A NEW APPROACH TO CIRCULAR UNIDIMENSIONAL SCALING

Li Chi Yin

A Thesis Submitted in Partial Fulfillment

of the Requirements for the Degree of

Master of Philosophy

in

Statistics

June 2002


香港中文大學
圖書館系統
06 MAR 2003
THE CHINESE UNIVERSITY OF HONG KONG

THE CHINESE UNIVERSITY OF HONG KONG

GRADUATE SCHOOL

The undersigned certify that we have read a thesis, entitled "A new approach to circular unidimensional scaling" submitted to the Graduate School by Li Chi-yin ( ) in partial fulfillment of the requirements for the degree of Master of Philosophy in Statistics. We recommend that it be accepted.

Prof. P.L. Leung,
Supervisor

Prof. P.S. Chan

Prof. T.S. Lau

Prof. Y.V. Hui,
External Examiner

DECLARATION

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institution of learning.

ACKNOWLEDGMENT

I am deeply indebted to my supervisor, Dr. Leung Pui Lam, for his generosity of encouragement and supervision. It is also a pleasure to express my gratitude to all the staff of the Department of Statistics for their kind assistance.

ABSTRACT

In this thesis, we proposed a method for finding a good starting values for the Guttman's updating algorithm and Pliner's smoothing algorithm in unidimensional scaling(UDS). We proved that when the interpoint distances are measured without errors, our new method can find the true solution in one iteration using Guttman's updating algorithm. Simulation studies are carried out and indicates that our new method provides a better solution as well as reducing the computing time. Furthermore, a new procedure for finding solution in circular unidimensional scaling (CDS) is proposed. In this method, objects are first arranged in a linear order as in UDS, and then the objects are restricted to locate on the circumference of a circle. Finally, extension of this method to data mining is also briefly discussed.

Keywords: Multidimensional scaling; Pliner's smoothing technique; Guttman's updating technique; Data mining

摘要

在這篇論文中，我們提出一個尋找良好初始值的方法，藉以應用在能處理一維尺度問題的 Guttman 更新算法和 Pliner 平滑算法上。我們證明了若量度距離時沒有出現誤差，使用我們的初始值去實行 Guttman 的更新算法可以在一次重複中得到真正的解。我們進行了一項模擬研究，這項研究顯示出使用我們的方法可以得到更好的解，並且減少了計算的時間。另外，我們提出了一個新方法去處理環狀一維尺度問題，這方法會先將物件排列成線性次序，然後再把它們的位置限制在一個圓周上。我們最後還會簡略地討論如何伸展這方法到數據提煉上。

關鍵字: 多維尺度; Pliner 平滑算法; Guttman 更新算法; 資料提煉

Contents

1 Introduction **1**

1.1 Multidimensional Scaling (MDS) 1

1.2 Unidimensional Scaling (UDS) 15

1.3 Circular Unidimensional Scaling (CDS) 17

1.4 The Goodness of fit of models 24

1.5 The admissible transformations of the MDS configuration 26

2 Computational Methods on MDS, UDS and CDS **29**

2.1 Classical Scaling 29

2.2 Guttman's updating algorithm and Pliner's smoothing algorithm . 36

2.3 Circular Unidimensional Scaling/Circumplex Model 43

3 A new algorithm for CDS **45**

3.1 Method of choosing a good starting value in Guttman's updating algorithm and Pliner's smoothing algorithm 46

3.2 A new approach for circular unidimensional scaling 54

3.3 Examples 62

3.3.1 Comparison of the new approach to existing method . . . 62

3.3.2 Illustrations of application to political data 64

4 Conclusion and Extensions **67**

A Figures and Tables **70**

B References **78**

List of Tables

1.1 Table of the distances between ten U.S. cities 3

1.2 The correlation matrix of 7 crime rates in 50 U.S. states 5

1.3 The dissimilarity matrix of 7 crime rates in 50 U.S. states 6

1.4 The dissimilarity matrix of 10 brands of soft drinks 8

1.5 The distance matrix of different brands of mobile phone 13

1.6 A proximity matrix for the 10 Morse code symbols 20

1.7 The Circumplex model applied by Pluchik and Conte(1994)to the 14 personality disorders 23

3.1 A proximity matrix for the eight Morse code symbols 50

3.2 A proximity matrix for twelve Second World War politicians . . . 65

A.1 The comparison of the stress value of Guttman's updating algorithm on data without noise (50 random starting values) 72

A.2 The comparison of the stress value of Guttman's updating algorithm and Pliner's smoothing algorithm for n=100 and $\sigma_\tau = 0.5$ (50 random starting values) 73

A.3 The comparison of the stress value of Guttman's updating algorithm and Pliner's smoothing algorithm for n=1000 and $\sigma_\tau = 0.5$ (50 random starting values) 74

A.4 The comparison of the mean number of iteration required by Guttman's updating algorithm and Pliner's smoothing algorithm for n = 100 (50 random starting values) 75

A.5 The comparison of the mean number of iteration required by Guttman's updating algorithm and Pliner's smoothing algorithm for n = 1000 (50 random starting values) 75

A.6 The stress value of the proposed algorithm for the data(in 100 trials) without and with error,$\sigma_\tau = 0.5$, n=20 76

A.7 The stress value of the proposed algorithm for the data (in 100 trials) without and with error,$\sigma_\tau = 0.5$, n=50 77

List of Figures

1.1 The locations of ten U.S. cities 4

1.2 The configuration of 10 U.S. cities constructed by Multidimensional Scaling 4

1.3 The two dimensional MDS representation of the correlations between 7 crime rates in 50 U.S. states 7

1.4 The 2-dimensional MDS representation of 10 brands of soft drinks 9

1.5 The unidimensional scaling representation of 7 crime rate in 50 states 16

1.6 The solution of classical scaling for Kendall's similarity matrix for 51 graves 19

1.7 The solution of classical scaling for 10 Morse Code symbols 21

1.8 The CDS representation of the similarity to the 14 personality disorder 24

3.1 A unidimensional scaling representation 51

3.2 A unidimensional scaling representation of Morse Code(by Pliner's smoothing algorithm) 51

3.3 Changing the configuration of the objects from circular to linear . 57

3.4 The choice of the position of 'g' 60

3.5 The CDS of the Morse Code data from Table 3.1, $\sigma_1 = 0.1338$ . . 60

3.6 The optimal representation of the Morse Code data from Table 3.1, $\sigma_1 = 0.1209$ 61

3.7 The CDS by Hubert, Arabie and Meulman's method for the Morse Code data from Table 1.6 63

3.8 The CDS by proposed method for the Morse Code data from Table 1.6 64

3.9 The MDS representation by classical scaling method for the similarity of the politicians from Table 3.2 66

3.10 The CDS representation by proposed method for the similarity of the politicians from Table 3.2 66

A.1 The box plots of the stress value given by the proposed method,(n=20, 100 trials, $\sigma_\tau = 0.5$) 70

A.2 The box plots of the stress value given by the proposed method,(n=50, 100 trials, $\sigma_\tau = 0.5$) 71

Chapter 1

Introduction

Multidimensional scaling (MDS) is a method used for the analysis of similarity or dissimilarity of a set of objects. One of the special case is the Unidimensional scaling (UDS), which represent the objects in one dimension. In the following, we will describe the idea and the use of MDS, UDS and the Circular Unidimensional scaling(CDS).

1.1 Multidimensional Scaling (MDS)

Multidimensional Scaling(MDS) is used to represent the configuration of objects by using the proximity among all pairs of objects. Such technique is useful in many disciplines, including physical, behavioral sciences and marketing research.

The configuration of the points by MDS can be represent geometrically on a map. It is much easier to interpret the structure of data by exploring the picture constructed by MDS method than just by looking at the proximity matrix. For example, we can find out which objects are similar(close on the map) and which objects are different(distant on the map) and thus we can cluster the data easily. In order to explain the idea, let us consider the following three examples.

Example 1.1.1 (Geographical Example)

Suppose we are given the distances between ten U.S. cities in Table 1.1 and we want to produce the coordinates of these cities on the map. The premier purpose of multidimensional scaling is to find out a representation of the cities such that the reproduce interpoint distance between the cities is as close as the observed one. We also want that the representation is as few dimension as possible. The two-dimensional MDS representation of the cities is shown in Figure 1.2. We will only show the result first, the detail of the procedure will be given in the later chapter.

We can see that the obtained locations of the cities in Figure 1.2 are very different from the truth in Figure 1.1. It is because the final constructed configuration of the cities may be rotated when the MDS procedure is applied, but it can be adjusted by reflecting the map along the horizontal direction such that the direction of West & East is reversed. Such transformation of MDS configuration

Table 1.1: Table of the distances between ten U.S. cities

	Atla	Chic	Denv	Hous	L.A.	Mia	N.Y.	S.F.	Seat	Wash
Atlanta	0									
Chicago	587	0								
Denver	1212	920	0							
Houston	701	940	879	0						
Los Angeles	1936	1745	831	1374	0					
Miami	604	1188	1726	968	2339	0				
New York	748	713	1631	1420	2451	1092	0			
San Francisco	2139	1858	949	1645	347	2594	2571	0		
Seattle	2182	1737	1021	1891	959	2734	2408	678	0	
Washington D.C.	543	597	1494	1220	2300	923	205	2442	2329	0

is distance invariant. Other transformations such as rotation and translation are also invariant to the relative interpoint distances between the objects. The invariance property of these transformations will be discussed later. Figure 1.2 show that the closest cities are Washington D.C. & New York and the most distant cities are Seattle & Miami. Also, the reader can observe directly from the map to know whether a particular city is located in East Coast or West Coast, such information cannot be given if we observe the distance matrix in Table 1.1 only.

Figure 1.1: The locations of ten U.S. cities

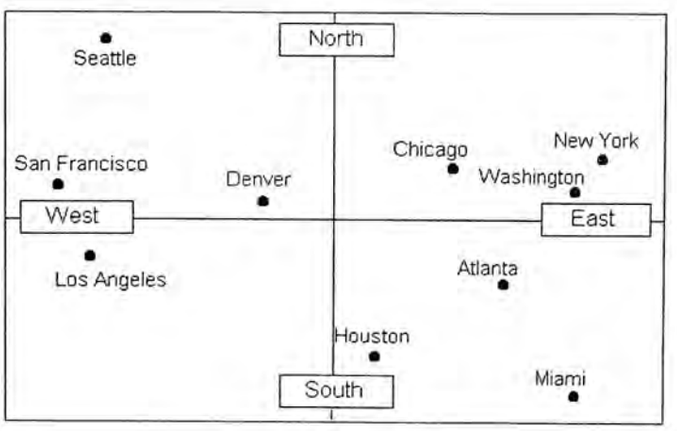
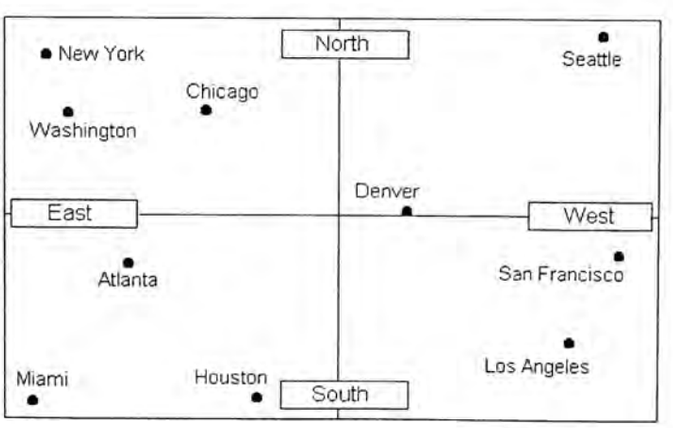


Figure 1.2: The configuration of 10 U.S. cities constructed by Multidimensional Scaling



Example 1.1.2 (Crime Rates Example)

Example 1.1.1 given above what MDS can do with the data, but the recovery of the positions of cities is the not the main purpose of MDS.(Indeed, map is not constructed by such method). As mentioned before, multidimensional scaling algorithm can be useful for psychological or behavioral science.

In 1970, the Bureau of the Census gave out the different crime rate in the 50 U.S. states in the issue of U.S. Statistical Abstract (Wilkinson(1990)). Borg and Groenen(1997) calculated the correlations between the seven crime rates and the result is given in Table 1.2. In this case, we produce the MDS representation of the informations(Figure 1.3). Note that the data are given in a correlation matrix, not the distance matrix, therefore we have to transform the correlation matrix into distance matrix, which is shown in Table 1.3., detail will be given later.

Table 1.2: The correlation matrix of 7 crime rates in 50 U.S. states

	murder	rape	robbery	assault	burglary	larceny	auto theft
murder	1.00						
rape	0.52	1.00					
robbery	0.34	0.55	1.00				
assault	0.81	0.70	0.56	1.00			
burglary	0.28	0.68	0.62	0.52	1.00		
larceny	0.06	0.60	0.44	0.32	0.80	1.00	
auto theft	0.11	0.44	0.62	0.33	0.70	0.55	1.00

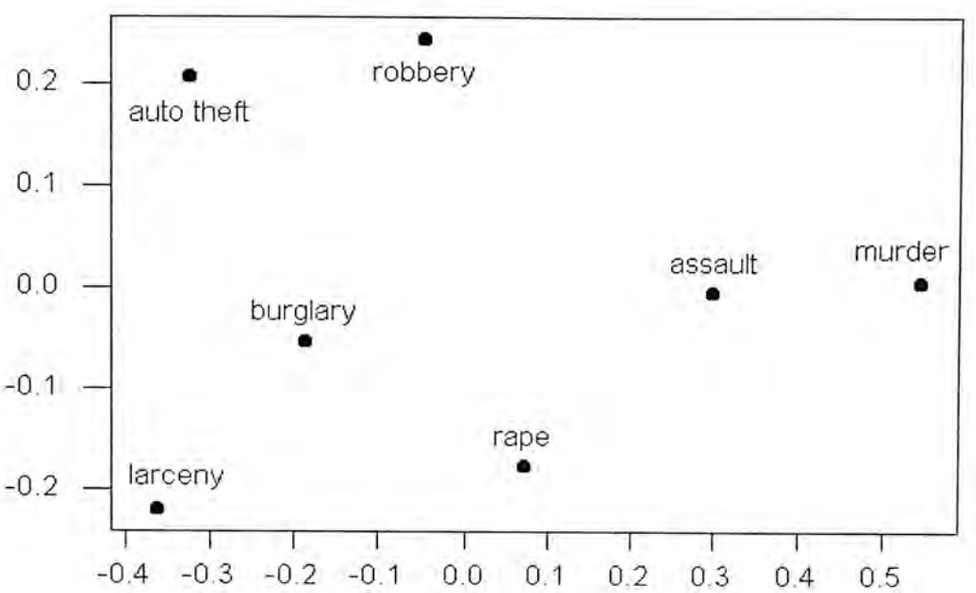
As this MDS representation is reflecting the correlations between the different crime rates, one can conclude that the longer the distance between two particular crime in the map, the lesser the correlations they have and vice versa. For example, a murderer is less likely to offend larceny than a burglar. Furthermore, we may consider these crimes as two dimensional, say 'violence' and 'possession',

Table 1.3: The dissimilarity matrix of 7 crime rates in 50 U.S. states

	murder	rape	robbery	assault	burglary	larceny	auto theft
murder	0.00						
rape	0.48	0.00					
robbery	0.66	0.45	0.00				
assault	0.19	0.30	0.44	0.00			
burglary	0.72	0.32	0.38	0.48	0.00		
larceny	0.94	0.40	0.56	0.68	0.20	0.00	
auto theft	0.89	0.56	0.38	0.67	0.30	0.45	0.00

by their structure. Murder, assault and rape which are close to each others can be grouped into 'violence' while the others are belong to 'possession'. Further research can then be done from this basic studies.

Figure 1.3: The two dimensional MDS representation of the correlations between 7 crime rates in 50 U.S. states



Example 1.1.3 (Marketing Example)

Knowing the buyers' perceptions & preference of the products is one of the important purpose of marketing research. The multidimensional scaling algorithm can help to find out the relationships between these behavioral data.

In order to test the perception of 60 graduate students of two universities on the similarity of ten brands of soft drinks, Green, Carmone and Smith(1989) asked them to finish a questionnaire and then use the information to calculate the dissimilarity of the soft drinks. The resulting matrix is given in Table 1.4 and a MDS representation of the data is plotted in Figure 1.4

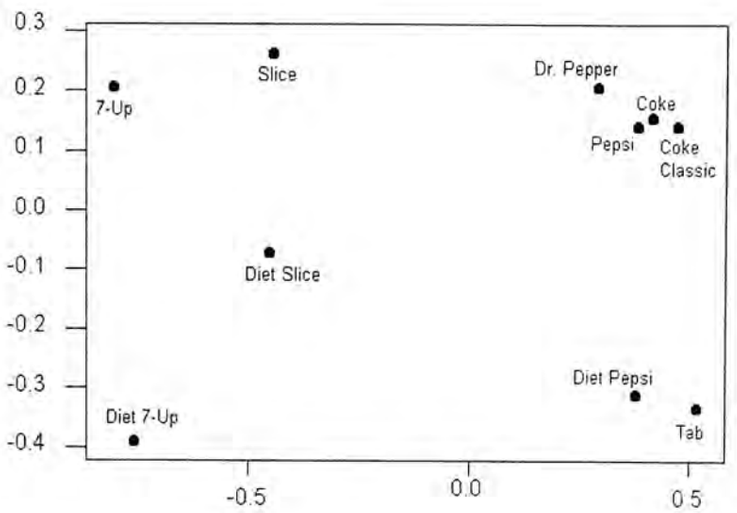
Table 1.4: The dissimilarity matrix of 10 brands of soft drinks

	P	C	CC	DP	DS	D7	DR	S	7	T
Pepsi	0									
Coke	1	0								
Coke Classic	3	2	0							
Diet Pepsi	7	9	8	0						
Diet Slice	27	28	32	22	0					
Diet 7-Up	41	42	43	29	13	0				
Dr. Pepper	18	17	19	20	30	40	0			
Slice	24	25	26	31	5	16	23	0		
7-Up	35	36	38	44	15	6	37	14	0	
Tab	12	11	10	4	33	34	21	39	45	0

We note from Figure 1.4 that several drinks are very close and some are very different from others. By interpreting the distance of the objects, we can consider the drinks in two dimensions: horizontal and vertical dimension. Green, Carmone and Smith(1989) stated that from other information of the questionnaire, the horizontal dimension can be interpreted as the flavor and the vertical dimension can be comprehended as the dietness of the soft drinks.

Therefore. Slice is said to be the most nondietetic and the flavor of Coke & Pepsi are very different from Slice & 7-Up. It is useful in analyzing the preference of the customers.

Figure 1.4: The 2-dimensional MDS representation of 10 brands of soft drinks



From the above three examples, we can see that there are more than one type of proximity measures between objects, but the most general defined measures are given as following:

Dissimilarity

Given the objects a and b, the dissimilarity measure, p_{ab} satisfies the following:

1. $p_{ab} \geq 0$ for all a, b

2. $p_{aa} = 0$ for all a

3. $p_{ab} = p_{ba}$ for all a, b

If two objects are very similar, the value of measure is small and vice versa. The most common example of this measure is the *Euclidean distance* and there are other distances such as *Minkowski distance* and *City-Block distance*.

Euclidean distance

The distance between two objects are calculated from their coordinates according to the Pythagorean formula:

$$d_{ab} = \sqrt{\sum_{i=1}^{m}(x_{ai} - x_{bi})^2} \tag{1.1}$$

where x_{ai} and x_{bi} are the coordinates of objects a and b respectively and m represent the dimension of the objects. Therefore, if $m = 2$, the equation will become:

$$d_{ab} = \sqrt{(x_a - x_b)^2 + (y_a - y_b)^2} \tag{1.2}$$

If we are given an $(m \times p)$ data matrix $\mathbf{X}$ with $(1 \times p)$ row vectors x'_1,x'_2...x'_m, the *Euclidean distance*, d_{ab} is:

$$d^2_{ab} = (x_a - x_b)'(x_a - x_b) \tag{1.3}$$

and then the $(n \times n)$ matrix of d^2_{ab} is called *squared Euclidean distance matrix*.

Although the *Euclidean distance* is easy to be calculated and understood, it has a disadvantage that it is sensitive to the scale of measurement. If the measurement scale for some variables are not common, variables with large scale may dominate the distance measurement. Therefore, the *standard Euclidean*

distance is suitable in such case. The *standard Euclidean distance* is given by

$$d_{ab}^2 = \sum_{i=1}^{m} (\frac{1}{s_j^2})(x_{ai} - x_{bi})^2 \tag{1.4}$$

or in matrix form

$$d_{ab}^2 = (x_a - x_b)' \mathbf{D}^{-1} (x_a - x_b) \tag{1.5}$$

where s_j^2, $j = 1, ..., m$ denotes the variance of X_j and $\mathbf{D}$ is the diagonal matrix with diagonal elements s_j^2

Minkowski distance

The *Minkowski distance* between objects a and b is the r^{th} root of the sum of the r^{th} power of the absolute value of the difference between their coordinates, where r is representing the dimension. If $r = 2$, then the distance between a and b is:

$$d_{ab} = (|x_a - x_b|^2 + |y_a - y_b|^2)^{1/2} \tag{1.6}$$

City-Block distance

The *City-Block distance* between objects i and j is the summation of the absolute coordinate differences between them :

$$d_{ij} = \sum_{a=1}^{m} |x_{ia} - x_{ja}| \tag{1.7}$$

where x_{ia} and x_{ja} are the scores of the attribute a of object i and j respectively and m is the number of interested attributes. In fact, *City-Block distance*

is the special case of *Minkowski distance* where $r = 1$.

Another type of function often used for calculating distance is to count the common elements over all attributes of the objects. Restle(1959) suggested the distance, d_{XY} between object X and Y as:

$$d_{XY} = g(X \cup Y) - g(X \cap Y) \tag{1.8}$$

where g is the measure function, $X \cup Y$ is the union of X & Y and $X \cap Y$ is the intersection of X & Y. One of the simple measure function g is the number of the elements in the set and thus the d_{XY} is the number the their noncommon features.

For example, Suppose a mobile phone company wants to know the opinions of the users on the similarity between different brands of mobile phone. A questionnaire is given to them and the users are asked to response to 10 questions for each brand using the following keys:

1. Strongly disagree
2. Disagree
3. Not Sure
4. Agree
5. Strong agree

The results are given below:

	Questions									
Brands	1	2	3	4	5	6	7	8	9	10
A	2	1	4	2	3	1	3	2	5	2
B	3	1	4	2	1	2	3	2	4	2
C	1	2	2	2	3	1	3	3	4	5
.	.	.	.	.	.	.	.	.	.	.

The range of d_{ij} is from zero to ten and the number of questions that Brand A and Brand B have different answers is 4, therefore $d_{AB} = 4$ and then $d_{AC} = 6$ & $d_{BC} = 7$. A distance matrix can be constructed as below:

Table 1.5: The distance matrix of different brands of mobile phone

Brands	A	B	C	.	.	.
A	0	4	6	.	.	.
B	4	0	7	.	.	.
C	6	7	0	.	.	.
.	.	.	.	.	.	

Similarity

If the measures of object a and b, p_{ab}, are the similarity measure, it satisfies

1. $0 \leq p_{ab} \leq 1$ for all a, b

2. $p_{aa} = 0$ for all a

3. $p_{ab} = p_{ba}$ for all a, b

The most common measure of similarity is the correlation coefficient, ρ, where the expression of correlation coefficient for a pair of variables, $\mathbf{X}$ and $\mathbf{Y}$ with N components is given by

$$\rho = \frac{\sum_{i=1}^{N}(x_i - \overline{x})(y_i - \overline{y})}{(\sum_{i=1}^{N}(x_i - \overline{x})^2)^{1/2}(\sum_{i=1}^{N}(y_i - \overline{y})^2)^{1/2}}. \tag{1.9}$$

We will only take the absolute value of it since the range of the coefficient is between -1 and 1 which make the value of p_{ab} has a probability that less zero.

Sometimes, we want to have the dissimilarity measure of the data, but we are only given the similarity information, for example, the correlation coefficient matrix. One way we can do is to transform the elements of the matrix. Recall Example 1.1.2, we can transform the data by subtracting all the elements of the matrix from one. Taking absolute value is need if the correlation coefficient in the matrix is negative. Then the transformation(in Table 1.3) will fulfill all the property of dissimilarity measure.

In the applications of the daily case, the properties of similarity and similarity measures mentioned above may be violated since in practice, the data usually contain noise. Such noisy data make the work of MDS become more complicated.

Meanwhile, although all the examples given in this section are two dimensional MDS, it is not rare that in some cases, the data are consideration of the dimension is involving the complexity during the explanations of the result. Therefore, high dimension MDS representation of data are usually not preferable. The simplest case is the one dimensional case of MDS, called unidimensional scaling and it will be discussed in the next section.

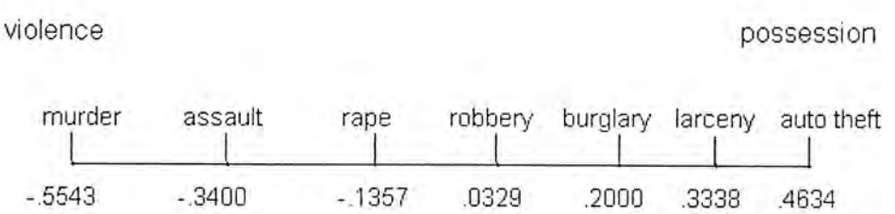
1.2 Unidimensional Scaling (UDS)

We have discussed that the multidimensional scaling is used to consider the objects in multidimensional space before. In this section, we will concern about the one-dimensional case of multidimensional scaling, the unidimensional scaling(UDS). It is applicable when objects are classified according to one property. For example, sometimes the social scientists or psychologists may want to measure a single personal characteristic only, then the UDS algorithm is appropriate for this situation. The unidimensional scaling algorithm is also suitable for interpreting the single dimension underlies preferential choice data or similarities data.

In the real cases of psychological studies, the objects are usually classified by two or more properties. If we apply unidimensional scaling to the multidimensional scaling case, some information will lose and thus the result will become less

accurate and reliable, but why the unidimensional scaling algorithm can still be wildly used for many cases? Let us recall the crime rate example in Section 1.1. This time, instead of considering the crimes as a multidimensional(two dimensional)case, we apply the UDS algorithm, the Pliner's smoothing algorithm(which the details will be discussed later) to the data set in Table 1.3 and then we have the UDS representation in Figure 1.5:

Figure 1.5: The unidimensional scaling representation of 7 crime rate in 50 states



Since the objects are restricted on a straight line, the representation cannot be compared directly with that in the MDS case. However, from the UDS representation, we can see that the three kind of crimes: murder, rape and assault, which are considered as 'violence' type crime in the multidimensional case are located closely on the negative side and the others: robbery, burglary, larceny and auto theft which are considered as 'possession' in MDS case are located on the opposite positive side. As a result, the same conclusion can also be drawn by the unidimensional scaling. Meanwhile, the clear separation between the two types of the crimes makes our interpretation easier than that in the MDS case.

In many cases, unidimensional scaling is a valuable alternative to the multidimensional scaling because, as shown in our example, the UDS representation is easier to apply and the result can easily be followed.

The other reason which make the unidimensional scaling applicable for multidimensional case is that although the higher the dimension we apply in MDS, the higher the precision of the results we can get, at the same time, the more difficult for explanation, we can imagine that it is not an easy task to explain to others about a five dimensional MDS representation of the objects. Furthermore, we don't know the suitable dimension we should used since we don't have the information about the objects and their properties. The unidimensional scaling can act as a prior information for researcher. Through the information given by this process, we can have a rough picture on the structure of the data before the later steps of the interpretation. Therefore, inspire of the less validity of unidimensional scaling, it is still adopted by researchers.

1.3 Circular Unidimensional Scaling (CDS)

Sometime we may face a proximity matrix that is not well patterned unidimensionally and the configurations of the objects may have a circular shape(as shown later). In such cases, the circular unidimensional scaling is suitable for such data sets. The Circular Unidimensional Scaling, as the name implied, is the

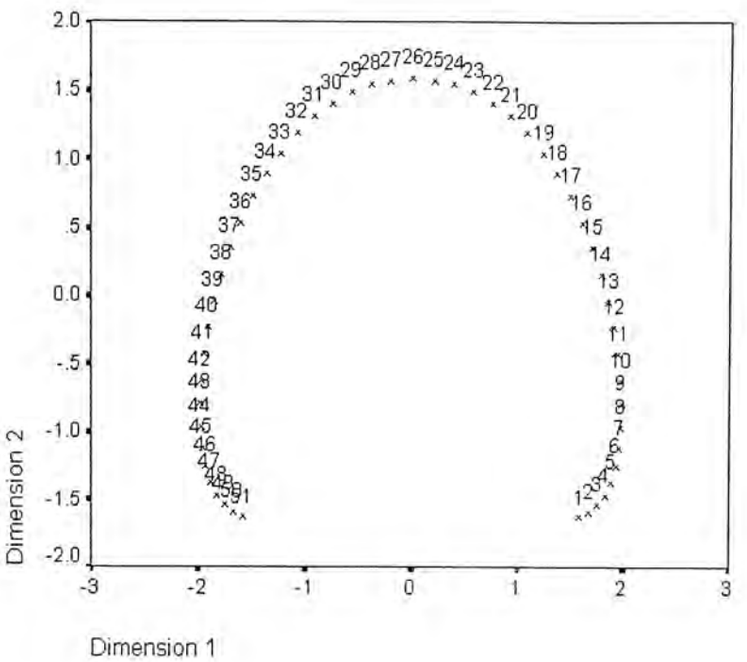
circular case of unidimensional scaling. It is usually applied for the interpretations of emotion or personality traits in psychology. Its aim is to find out the coordinates of n objects and identify the order of the them.

Guttman(1954) stated that variables can be ordered around a circular continuum according to differences of their properties and used the term 'circumplex' to represent these idea where 'circumplex' is the short form of 'circular continuum of complexity'. In some cases, when we apply classical multidimensional scaling to the dissimilarity matrix, the configuration given out may be either circular or horseshoe shape. Kendall(1970) created an artifical similarity matrix for 51 hypothetical graves in chronological order, separated by equal time-interval as follows:

$$s_{ij} = \begin{cases} 9 \quad \text{if} \quad i = j \\ 8 \quad \text{if} \quad 1 \leq |i-j| \leq 3 \\ . \\ . \\ 1 \quad \text{if} \quad 22 \leq |i-j| \leq 24 \\ 0 \quad \text{if} \quad \geq 25 \end{cases}$$

A dissimilarity matrix is constructed by calculating $d_{ij} = (s_{ii} + s_{jj} - 2s_{ij})^{1/2}$. Applying classical scaling, the configuration is shown in Figure 1.6. A clear horseshoe shape is shown because the distances between all of the object i and j are the same when $|i-j| \geq 25$.

Figure 1.6: The solution of classical scaling for Kendall's similarity matrix for 51 graves



The shape of the MDS configuration given by the 10 Morse Code symbols representing the first ten digits (data from Rothkoft(1957) in Table 1.6) is also circular. This can be shown in Figure 1.7

According to Guttman, the order of the variables in circumplex model is not identified by the their ranking from highest to lowest. Therefore, there is no beginning and end in circular order. Such operation is suitable for any similarity coefficients, e.g. correlation coefficients. For a particular variable, the nearest neighboring is recognized to have the highest correlation with the focus. The correlations diminish gently when one moves clockwise or counter-clockwise away and up to the point π rad from it.(they are interpreted to have -1.0 correlation

Table 1.6: A proximity matrix for the 10 Morse code symbols

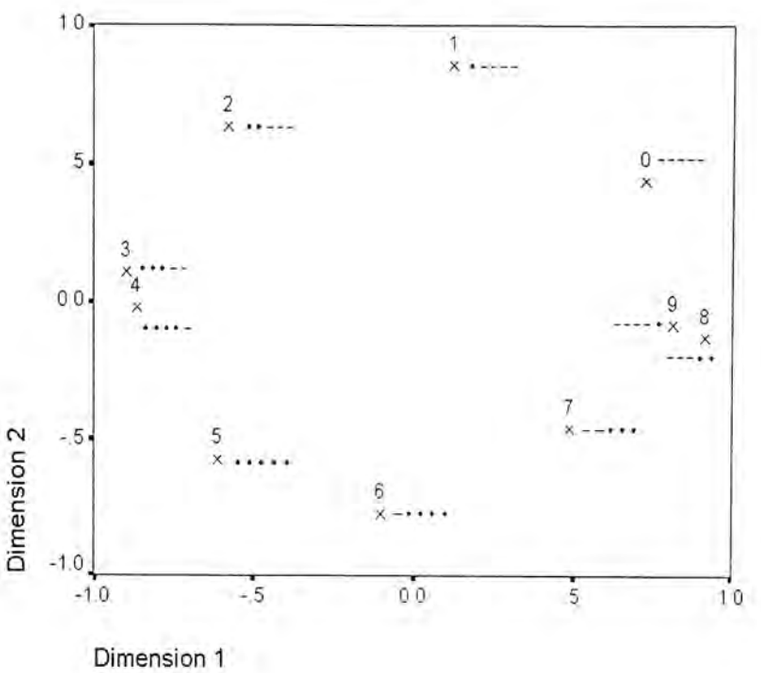
Symbol	0	1	2	3	4	5	6	7	8	9
0:- - - - -	0.00									
1:• - - - -	0.75	0.00								
2:• • - - -	1.69	0.82	0.00							
3:• • • - -	1.87	1.54	1.25	0.00						
4:• • • • -	1.76	1.85	1.47	0.89	0.00					
5:• • • • •	1.77	1.72	1.33	1.32	1.41	0.00				
6:- • • • •	1.59	1.51	1.66	1.53	1.64	0.70	0.00			
7:- - • • •	1.26	1.50	1.57	1.74	1.81	1.56	0.70	0.00		
8:- - - • •	0.86	1.45	1.83	1.85	1.90	1.84	1.38	0.83	0.00	
9:- - - - •	0.95	1.63	1.81	1.86	1.90	1.64	1.70	1.22	0.41	0.00

at this situation). After that, the correlations increase again as it returns to the initial variable.

Similar to MDS representation, the circular unidimensional scaling can be rotated clockwisely or counter-clockwisely since the relationships between the objects in the circle is invariant to any changes of axes. Therefore, base on the result of other studies, for example, factor analysis, we can set the axes and rotate the circle arbitrarily for better understanding of the data.

As the configuration of the objects in Circular Unidimensional Scaling is restricted on the continuum of a circle, if the dissimilarity used for the data is the their interpoint distance, we will only consider the minimum distance between them. Thus for two objects i and j their interpoint distance, d_{ij} is defined in the

Figure 1.7: The solution of classical scaling for 10 Morse Code symbols



following way:

$$d_{ij} = min\{|x_j - x_i|, l - |x_j - x_i|\} \tag{1.10}$$

where x_i and x_j are the coordinates of object i and j respectively and l is the circumference of the circle.

Besides using the interpoint distance, we can measure the dissimilarity between the objects in term of degree. The difference in degree of object i and j, θ_{ij} is defined as:

$$\theta_{ij} = min\{|\theta_j - \theta_i|, 2\pi - |\theta_j - \theta_i|\} \tag{1.11}$$

where θ_i and θ_j are the degrees of object i and j respectively from 0 rad.

For fixed permutation of the objects, $\varphi(i) = j$, $1 \leq i, j \leq n$, which means that the i^{th} object is object j, the circumference of the circle, l is given by:

$$l = d_{\varphi(1)\varphi(2)} + d_{\varphi(2)\varphi(3)} + ... + d_{\varphi(j)\varphi(j+1)} + ... + d_{\varphi(n)\varphi(1)} \tag{1.12}$$

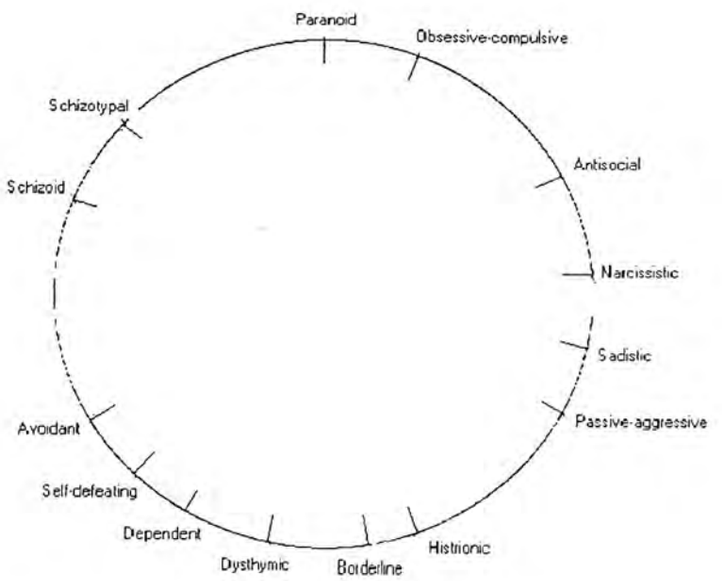
Since it is impossible to measure the distances between objects without any error, we cannot find out the true value of the circumference of the circle. However, it is not a very serious problem when the main purpose of our work is to find out the relationships between the objects, not the value of l. Also, although the configuration of the data is circular, the circumflex model is consider as a one dimensional representation. Plutchik and Conte(1994) carry out a study about the personality disorders of human being. Table 1.7 give out their circumplex model and the circular representation of the objects is shown in Figure 1.8. This result is harmonic to the diagnoses of the structure of these traits in many former studies.

Table 1.7: The Circumplex model applied by Pluchik and Conte(1994)to the 14 personality disorders

Disorders	Angle(degrees)
Paranoid	0
Obsessive-compulsive	20
Antisocial	60
Narcissistic	85
Sadistic	102
Passive-aggressive	117
Histrionic	160
Borderline	170
Dysthymic	192
Dependent	210
Self-defeating	225
Avoidant	240
Schizoid	292
Schizo typal	312

Similar to the case of UDS, as the representation of data in CDS is restricted on the continuum of the circle, it may be less precise when we apply such model to the data rather than using MDS, but one of the advantages of using CDS or circumplex model is that it can represent the similarity and polarity of objects clearly. Therefore, there are many researchers introduce such approach to their studies.

Figure 1.8: The CDS representation of the similarity to the 14 personality disorder



1.4 The Goodness of fit of models

In MDS, UDS and Circumplex/CDS models, we map the proximity (similarity or dissimilarity), p_{ij} of the data into the distance of configuration X by a representation function, $f : p_{ij} \rightarrow d_{ij}$. No matter for which models, what a sought is a configuration whose distances, d_{ij} is as close as possible to $f(p_{ij})$. If the discrepancy between them is too large, the model may not be reliable enough. The closeness can be measured by a loss function, which is a mathematical expression that representing the errors, $e_{ij} = f(p_{ij}) - d_{ij}(X)$, over all pairs i, j. That means if the observed data is fitted well in the model, the value of e_{ij} will be small.

There are many defined loss function and the most common lost function is *Stress function*. The *Stress function* is summation of the squared error of repre-

sentation, e_{ij}^2 over all pairs of i, j, where the squared error, e_{ij}^2 is defined by

$$e_{ij}^2 = [f(p_{ij}) - d_{ij}]^2 \tag{1.13}$$

Thus the raw *Stress* is given by

$$\sigma_r^2 = \sum_{i<j}[f(p_{ij}) - d_{ij}(X)]^2 \tag{1.14}$$

However, that above definition is very sensitive to the scale of data. For example, if the dissimilarity are Euclidean distances between cities in kilometers and the raw *Stress* in MDS model is given by $\sigma_r(X_k)^2 = 0.1$. Suppose we measure the distance again but use meters expression instead, we get $\sigma_r(X_m)^2 = 100000$, which is 1000000 times larger, but we will not conclude that X_k is fitted better than X_m. Therefore, a small value of σ_r^2 may not imply that the fitting is good this time.

In order to avoid this problem, a normed *Stress function* is defined as following

$$\sigma^2 = \frac{\sigma_r^2(X)}{\sum_{i<j} d_{ij}^2(X)} = \frac{\sum_{i<j}[f(p_{ij}) - d_{ij}(X)]^2}{\sum_{i<j} d_{ij}^2(X)} \tag{1.15}$$

or by taking square root of σ^2, define a value known as *Stress - 1* (Kruskal, 1964a):

$$Stress - 1 = \sigma_1 = \sqrt{\frac{\sum_{i<j}[f(p_{ij}) - d_{ij}(X)]^2}{\sum_{i<j} d_{ij}^2(X)}} \tag{1.16}$$

The optimal configuration X is determined by minimizing the value of *Stress* and the proximity, $f(p_{ij})$ are the 'approximated distance', $\hat{d}_{ij}$ in many cases.

1.5 The admissible transformations of the MDS configuration

Transformations of MDS representation are usually applied in order to make the interpretation easier, but it is important that the transformation should be restricted to admissible one,i.e.,it will not change the interpoint distances between the objects.

The transformations of the MDS configuration which leave the distance invariance is known as rigid motions. It is obvious that rotation and reflection are rigid motions. Another common form of transformation, translation, is also invariant to distances. Translation is a displacement of the configuration with respect to a fixed point, for example, one can shift all cities the same distance to right in Figure 1.6. The invariant property of the three transformations are given below:

Rotation

Let $\mathbf{X}$ be a $n \times m$ coordinates matrix, it can be shown that the matrix of squared distance, denoted by $\mathbf{D}^{(2)}(\mathbf{X})$ can be expressed $\mathbf{D}^{(2)} = d1' - 2\mathbf{X}\mathbf{X}' + 1d'$ where d is the vector containing the diagonal elements of $\mathbf{X}\mathbf{X}'$(the detail explanation is given in Chapter 2.1). The new rotated coordinates matrix can then be produced by $\mathbf{X}^* = \mathbf{X}\mathbf{H}$ where $\mathbf{H}$ is a orthonormal matrix. Then the new

distance matrix is

$$\mathbf{D}^{*2}(\mathbf{X}) = d^*1' - 2\mathbf{X}^*\mathbf{X}^{*\prime} + 1d^{*\prime} \tag{1.17}$$

Since $\mathbf{X}^*\mathbf{X}^{*\prime} = (\mathbf{XH})(\mathbf{XH})' = \mathbf{XHH}'\mathbf{X}' = \mathbf{XX}'$. Also, the vector d^* comes from the diagonal elements of $\mathbf{X}^*\mathbf{X}^{*\prime}$, therefore $d^* = d$ and thus $\mathbf{D}^{*2}(\mathbf{X}) = \mathbf{D}^2(\mathbf{X})$.

Reflection

With the same setting of Rotation. Let $\mathbf{X}^* = -\mathbf{X}$, then $\mathbf{X}^*\mathbf{X}^{*\prime} = (-\mathbf{X})(-\mathbf{X})' = \mathbf{XX}'$ and so $d^* = d$. The distance matrix is then remain unchanged after the reflection

Translation

The elements of the squared distance matrix, $\mathbf{D^2}(\mathbf{X})$ are given by

$$d_{ab}^2 = \sum_{j=1}^{n} X_{bj}^2 - 2\sum_{j=1}^{n} X_{aj}X_{bj} + \sum_{j=1}^{n} X_{aj}^2 \tag{1.18}$$

Let $X^* = X + C$, then the elements of D^{*2} is given by

$$\begin{aligned} {d_{ab}^*}^2 &= \sum_{j=1}^{n} {X_{bj}^*}^2 - 2\sum_{j=1}^{n} X_{aj}^*X_{bj}^* + \sum_{j=1}^{n} {X_{aj}^*}^2 \\ &= \sum_{j=1}^{n}(X_{bj}+C)^2 - 2\sum_{j=1}^{n}(X_{aj}+C)(X_{bj}+C) + \sum_{j=1}^{n}(X_{aj}+C)^2 \\ &= \sum_{j=1}^{n} X_{bj}^2 - 2\sum_{j=1}^{n} X_{aj}X_{bj} + \sum_{j=1}^{n} X_{aj}^2 \\ &= d_{ab}^2 \end{aligned}$$

Away from rigid motions, there are another admissible transformations, dilation, which is the enlargements or reductions of the configuration, the ratios of

the interpoint distances between objects remain unchanged. For both transformations, rigid motions and dilation, the value of objective function, Stress-1 in equation(1.16), as shown before is also invariant to this change. The shape of the whole picture will not be affected.

Chapter 2

Computational Methods on MDS, UDS and CDS

In last chapter, we presented the basic ideas and the applications of Multidimensional scaling, Unidimensional scaling and Circular Unidimensional Scaling/Circumplex model. With the purpose of minimizing the loss function(Stress function), many computational methods are proposed. In the following, we will describe some existing methods and illustrate how they can work with real data.

2.1 Classical Scaling

The first practical method for MDS is known as classical scaling. In this method, the dissimilarities are assumed to be distances and used for finding the

coordinates of objects. Therefore, before expressing the algorithm of it, we will show how to find the squared Euclidean distance between all points first.

Recall from Section 1.1 that, the squared Euclidean distance between object i and j with n dimensions is defined by

$$\begin{aligned} d_{ij}^2 &= \sum_{a=1}^{n}(x_{ia} - x_{ja})^2 \\ &= \sum_{a=1}^{n}(x_{ia}^2 - 2x_{ia}x_{ja} + x_{ja}^2) \end{aligned}$$

Borg & Groenen(1997) showed the method of calculating the matrix of squared distance. For simplicity, a coordinates matrix of three points in two dimensions, denoted by $\mathbf{X}$, is used, and then the squared distance in matrix form, denoted by $\mathbf{D^{(2)}(X)}$ is

$$\begin{aligned} \mathbf{D^{(2)}(X)} &= \begin{bmatrix} 0 & d_{12}^2 & d_{13}^2 \\ d_{12}^2 & 0 & d_{23}^2 \\ d_{13}^2 & d_{23}^2 & 0 \end{bmatrix} \\ &= \sum_{a=1}^{n} \begin{bmatrix} x_{1a}^2 & x_{1a}^2 & x_{1a}^2 \\ x_{2a}^2 & x_{2a}^2 & x_{2a}^2 \\ x_{3a}^2 & x_{3a}^2 & x_{3a}^2 \end{bmatrix} - 2\sum_{a=1}^{n} \begin{bmatrix} x_{1a}x_{1a} & x_{1a}x_{2a} & x_{1a}x_{3a} \\ x_{2a}x_{1a} & x_{2a}x_{2a} & x_{2a}x_{3a} \\ x_{3a}x_{1a} & x_{3a}x_{2a} & x_{3a}x_{3a} \end{bmatrix} + \\ &\quad \sum_{a=1}^{n} \begin{bmatrix} x_{1a}^2 & x_{2a}^2 & x_{3a}^2 \\ x_{1a}^2 & x_{2a}^2 & x_{3a}^2 \\ x_{1a}^2 & x_{2a}^2 & x_{3a}^2 \end{bmatrix} \\ &= d1' - 2\sum_{a=1}^{n} x_a x_a' + 1d' = d1' - 2\mathbf{X}\mathbf{X}' + 1d' \end{aligned} \tag{2.1}$$

where x_a is the column a of X and d is the vector containing the diagonal elements of XX'. For convenience, let the matrix $\mathbf{C} = \mathbf{XX'}$ and $\mathbf{C}$ is called a scalar product matrix. To illustrate the procedures, an example is shown as below:

Suppose

$$\mathbf{X} = \begin{bmatrix} 0 & 0 \\ 1 & 2 \\ 3 & 4 \end{bmatrix}$$

Then

$$\mathbf{C} = \mathbf{XX'} = \begin{bmatrix} 0 & 0 \\ 1 & 2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} 0 & 1 & 3 \\ 0 & 2 & 4 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 5 & 11 \\ 0 & 11 & 25 \end{bmatrix}$$

And

$$d' = \begin{pmatrix} 0 & 5 & 25 \end{pmatrix}$$

Thus

$$\mathbf{D^{(2)}(X)} = \begin{bmatrix} 0 & 0 & 0 \\ 5 & 5 & 5 \\ 25 & 25 & 25 \end{bmatrix} - 2 \begin{bmatrix} 0 & 0 & 0 \\ 0 & 5 & 11 \\ 0 & 11 & 25 \end{bmatrix} + \begin{bmatrix} 0 & 5 & 25 \\ 0 & 5 & 25 \\ 0 & 5 & 25 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 5 & 25 \\ 5 & 0 & 8 \\ 25 & 8 & 0 \end{bmatrix}$$

By taking the square root of all the elements, we have

$$\mathbf{D}(\mathbf{X}) = \begin{bmatrix} 0 & \sqrt{5} & 5 \\ \sqrt{5} & 0 & \sqrt{8} \\ 5 & \sqrt{8} & 0 \end{bmatrix}$$

If we multiply a centering matrix $\mathbf{K} = \mathbf{I} - n^{-1}11'$ to both sides of $\mathbf{D}^{(2)}$ and then multiply the factor $-\frac{1}{2}$ to the result, we can decompose it into three parts,

$$\begin{aligned} -\frac{1}{2}\mathbf{K}\mathbf{D}^{(2)}\mathbf{K} &= -\frac{1}{2}\mathbf{K}(d1' - 2\mathbf{X}\mathbf{X}' + 1d')\mathbf{K} \\ &= -\frac{1}{2}\mathbf{K}d1'\mathbf{K} - \frac{1}{2}\mathbf{K}1d'\mathbf{K} + \mathbf{K}\mathbf{X}\mathbf{X}'\mathbf{K} \\ &= \mathbf{C} \end{aligned}$$

Since $1'\mathbf{K}$ is a zero vector, the first two terms are equal to zero. Meanwhile, because of the invariant property of the interpoint distances, we assume that $\mathbf{X}$ is column centered such that $\mathbf{K}\mathbf{X} = \mathbf{X}$, thus $\mathbf{K}\mathbf{X}\mathbf{X}'\mathbf{K} = \mathbf{X}\mathbf{X}'$ and the above result are then proved. This operation is called double centering. The procedures of classical scaling are developed from this method and the MDS coordinates is found by doing the eigendecomposition of $\mathbf{C}$, where $\mathbf{C} = \mathbf{P}\mathbf{\Lambda}\mathbf{P}' = (\mathbf{P}\mathbf{\Lambda}^{\frac{1}{2}})(\mathbf{P}\mathbf{\Lambda}^{\frac{1}{2}})'$. However, the squared distance matrix $\mathbf{D}^{(2)}(\mathbf{X})$ is replaced by the squared dissimilarity matrix, $\mathbf{\Delta}^{(2)}$. The algorithm for classical scaling is shown by the following

steps:

1. Calculate the squared dissimilarity matrix, $\mathbf{\Delta}^{(2)}$

2. Calculate $\mathbf{C}_{\mathbf{\Delta}}$ by applying double centering, i.e. $\mathbf{C}_{\mathbf{\Delta}} = -\frac{1}{2}\mathbf{K}\mathbf{\Delta}^{(2)}\mathbf{K}$.

3. Compute the eigendecomposition of $\mathbf{C}_{\mathbf{\Delta}} = \mathbf{P}\mathbf{\Lambda}\mathbf{P}'$.

4. Define $\mathbf{\Lambda}^*$ as a $k \times k$ diagonal matrix where the elements are the first k positive eigenvalues and then define $\mathbf{P}^*$ as the matrix containing the corresponding eigenvector of $\mathbf{\Lambda}^*$. The value of k is the dimensionality of the solution.

5. Define the coordinates matrix of classical scaling by

$$\mathbf{X} = \mathbf{P}^*\mathbf{\Lambda}^{*\frac{1}{2}}$$

No matter for any procedures, what we want to do is to minimize the loss function. The loss function of using $\mathbf{\Delta}$ instead of $\mathbf{D}$ is given by

$$\begin{aligned} \mathbf{L}(\mathbf{X}) &= || -\frac{\mathbf{1}}{\mathbf{2}}\mathbf{K}[\mathbf{D}^{(2)}(\mathbf{X}) - \mathbf{\Delta}^{\mathbf{2}}]\mathbf{K}||^2 \\ &= ||\mathbf{X}\mathbf{X}' + \frac{\mathbf{1}}{\mathbf{2}}\mathbf{K}\mathbf{\Delta}^{\mathbf{2}}\mathbf{K}||^2 \\ &= ||\mathbf{X}\mathbf{X}' - \mathbf{C}_{\mathbf{\Delta}}||^2 \end{aligned} \tag{2.2}$$

Gower(1966) showed that choosing the classical solution will minimize that

loss function. In classical scaling, only the positive eigenvalues are used and the negative values are ignored as errors.

To demonstrate the above procedures, we will apply it to the data set in Table 1.3. For simplicity, we just use the data of the first four crimes rate, 'murder', 'rape' , 'robbery' and 'assault' for this example, so $n = 4$:

$$\boldsymbol{\Delta} = \begin{bmatrix} 0.00 & 0.48 & 0.66 & 0.19 \\ & 0.00 & 0.45 & 0.30 \\ & & 0.00 & 0.44 \\ & & & 0.00 \end{bmatrix}$$

Applying double centering,

$$\mathbf{K} = \mathbf{I} - n^{-1}11' = \begin{bmatrix} 0.75 & -0.25 & -0.25 & -0.25 \\ -0.25 & 0.75 & -0.25 & -0.25 \\ -0.25 & -0.25 & 0.75 & -0.25 \\ -0.25 & -0.25 & -0.25 & 0.75 \end{bmatrix}$$

$$\mathbf{C}_{\boldsymbol{\Delta}} = -\frac{\mathbf{1}}{\mathbf{2}}\mathbf{K}\boldsymbol{\Delta}^{(2)}\mathbf{K} = \begin{bmatrix} 0.101 & -0.036 & -0.100 & -0.035 \\ & 0.057 & -0.006 & -0.014 \\ & & 0.134 & -0.027 \\ & & & 0.006 \end{bmatrix}$$

By eigenvalue decomposition of C_Δ, we get

$$\mathbf{P} = \begin{bmatrix} -0.651 & -0.502 & 0.500 & 0.272 \\ 0.126 & -0.125 & 0.500 & -0.848 \\ 0.722 & -0.204 & 0.500 & 0.432 \\ -0.198 & 0.831 & 0.500 & 0.143 \end{bmatrix}$$

$$\mathbf{\Lambda} = \begin{bmatrix} 0.230 & 0.000 & 0.000 & 0.000 \\ 0.000 & -0.007 & 0.000 & 0.000 \\ 0.000 & 0.000 & -0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.074 \end{bmatrix}$$

The final solution of the coordinates of the crimes is given by

$$\mathbf{X} = \mathbf{P}^*\mathbf{\Lambda}^{*\frac{1}{2}} = \begin{bmatrix} -0.651 & 0.272 \\ 0.126 & -0.848 \\ 0.722 & 0.432 \\ -0.198 & 0.143 \end{bmatrix} \begin{bmatrix} 0.480 & 0.000 \\ 0.000 & 0.271 \end{bmatrix}$$

$$= \begin{bmatrix} -0.312 & 0.074 \\ 0.061 & -0.230 \\ 0.347 & 0.117 \\ -0.095 & 0.039 \end{bmatrix}$$

The classical scaling method also work for large n and the graphs in Figure 1.3 is the plot of all 7 kinds of crime rates which the coordinates are produced by this method.

When classical scaling is adopted, the final solution may be transformed, for example, rotated, reflected or translated, but the interpoint distance of the objects remain unchanged. Since classical scaling requiring no iterations and the result can easily be interpreted with the help of plotting graphs, it is commonly used by researchers.

2.2 Guttman's updating algorithm and Pliner's smoothing algorithm

As indicated before, the unidimensional scaling problem is the one dimensional case of MDS(i.e. $m = 1$). In this section, we are going to introduce two existing methods, Guttman's updating algorithm and Pliner's smoothing algorithm for dealing with the UDS problem. The objective of the two methods is to place the objects along a straight line such that the loss function, $\sigma(X)$ is minimized, where:

$$\sigma(X) = \sum_{i<j} (d_{ij} - |x_i - x_j|)^2 \tag{2.3}$$

However, the function, $\sigma(X)$ usually has many local minima which make the task of minimization becomes more difficult. The detail of the two procedures are shown below:

Guttman's updating algorithm

Consider the expression of loss function, the term $|x_i - x_j|$ can be written as $(x_i - x_j)sign(x_i - x_j)$ where

$$sign(x_i - x_j) = \begin{cases} 1 & \text{for } x_i > x_j \\ 0 & \text{for } x_i = x_j \\ -1 & \text{for } x_i < x_j \end{cases} \tag{2.4}$$

Therefore, $\sigma(X)$ can be express as

$$\sigma(X) = \sum_{i<j} [d_{ij} - (x_i - x_j)sign(x_i - x_j)]^2 \tag{2.5}$$

Without loss of generality, we imposed that $\sum_{i=1}^{n} x_i = 0$. Pliner(1984) show that X is a local minimum of $\sigma(X)$ for all $k = 1, ..., n$ when

$$x_k = \frac{1}{n} \sum_{j=1}^{n} d_{ij} sign(x_k - x_j) \tag{2.6}$$

and

$$x_k \neq x_j \quad \text{for } k \neq j$$

An algorithm based on this result was derived by Guttman(1968). He proposed that at the $(r+1)^{th}$ iteration:

$$x_k^{r+1} = \frac{1}{n} \sum_{j=1}^{n} d_{kj} sign(x_k^r - x_j^r) \tag{2.7}$$

for $k = 1, ..., n$. The starting value of X can be any random vector. The convergence to a stationary point in finite steps is proved by de Leeuw and Heiser(1980).

Based on this algorithm , Pliner(1984) suggested a additional step to it. It is the replacement of $sign(x_k^r - x_j^r) = 0$ by $+1$ or -1 in the case of $x_k^r = x_j^r$. i.e. if $sign(x_k^r - x_j^r)$ is replaced by $+1$, -1 will be used to replace $sign(x_j^r - x_k^r)$ and vice versa. After the modification, the trapping at the stationary points can be prevent and the whole algorithm can converge after regular iterations. Consider the simple example:

Suppose $n = 3$, and the coordinates of the three objects are $y_1 = 4$, $y_2 = 10$ and $y_3 = 12$, then the distance matrix is given by

$$d_{ij} = \begin{bmatrix} 0 & 6 & 8 \\ 6 & 0 & 2 \\ 8 & 2 & 0 \end{bmatrix}$$

Applying the Guttman's updating algorithm, a vector $x^{[0]} = \begin{pmatrix} 1 & 2 & 3 \end{pmatrix}$ is randomly chosen as a starting value, then

$$\begin{aligned} x_1^{[1]} &= \frac{1}{3}\sum_{j=1}^{3} d_{1j}sign(1 - x_j) = \frac{-14}{3} \\ x_2^{[1]} &= \frac{1}{3}\sum_{j=1}^{3} d_{2j}sign(2 - x_j) = \frac{4}{3} \\ x_3^{[1]} &= \frac{1}{3}\sum_{j=1}^{3} d_{3j}sign(3 - x_j) = \frac{10}{3} \end{aligned}$$

The loss function of the result is zero this time, hence, the coordinates of the objects are recovered completely by this method after one iteration.

Since the solution may not be found with one iteration in most cases, the steps for finding the coordinates of objects are:

1. Use $X^{[0]}$(can be random or not) as a starting configuration

2. Set r = 1, Computer $\sigma(X^{[r-1]})$

3. Until $\sigma(X^{[r-1]}) - \sigma(X^{[r]}) < \epsilon$ or r = maximum number of iterations:

   Computer Guttman's updating algorithm $X^{[r+1]}$,

   r = r + 1

4. The final solution, $Z = X^{[r]}$

In the above procedures, ϵ is a very small positive constant and the purpose of the stopping criterion $\sigma(X^{[r-1]}) - \sigma(X^{[r]}) < \epsilon$ is to ensure that $X^{[r]}$ is a minimum point of $\sigma(X)$. It is shown in Theorem 2.1 that the output of Guttman's updating algorithm is centered.

Theorem 2.1. *The final coordinates obtained from Guttman's updating algorithm are centered (i.e.*$\sum_{i=1}^{n} z_i = \sum_{i=1}^{n} x_i^{[r]} = 0$*), even if the starting value is not centered.*

Proof

$$\sum_{i=1}^{n} z_i = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij} sign(x_i^{[r]} - x_j^{[r]})$$

Since $d_{ij} = d_{ji}$ and $sign(x_i - x_j) = -sign(x_j - x_i)$

$$\sum_{i=1}^{n} z_i = \frac{1}{n} \left[\sum_{i<j}^{n} d_{ij} - \sum_{j<i}^{n} d_{ij} \right] = 0 \tag{2.8}$$

There is a problem of using Guttman's updating algorithm that Z may only be the local minimum of $\sigma(X)$, but not the global one. In general, in many UDS problems, this updating method cannot find out the globally optimal solution because the objective function have local minima and the solution is also very sensitive to the choice of starting value. In order to improve this situation, a new method of choosing starting value instead of random one will be proposed in the next chapter, but we ill introduce the "smoothing algorithm" proposed by Pliner(1996) in the following first.

Pliner's smoothing algorithm

If a function, $f(x)$ is multiextremal(has many local minima), it is difficult for us to find out the global minimum point by equating the derivatives of $f(x)$ to zero. Yudin(1965) suggested an replacement of $f(x)$ by $f_\epsilon(x)$ where

$$f_\epsilon(x) = \frac{1}{2\epsilon}\int_{x-\epsilon}^{x+\epsilon} f(y)dy \tag{2.9}$$

The expression $f_\epsilon(x)$ is the mean value of $f(y)$ over the interval $[x-\epsilon, x+\epsilon]$ and $f_\epsilon(x)$ is minimized instead of $f(x)$ so that the original function is replaced by the significantly smoother one. The global minimum of $f_\epsilon(x)$ is close to that of $f(x)$ and $f_\epsilon(x)$ is closer to $f(x)$ if ϵ is smaller. Base on this idea, Pliner(1996) proposed a smoothing approach to the local minimum problem in UDS. The loss function

is given by:

$$\sigma_\epsilon(X) = \sum_{i<j} \left[d_{ij}^2 + (x_i - x_j)^2 - 2 d_{ij} g_\epsilon(x_i - x_j) \right] \tag{2.10}$$

with

$$g_\epsilon(t) = \begin{cases} t^2(3\epsilon - |t|)/3\epsilon^2 + \epsilon/3 & \text{if } |t| < \epsilon \\ |t| & \text{if } |t| \geq \epsilon \end{cases}$$

After equating the partial derivatives of $\sigma_\epsilon(X)$ to zero, we get

$$x_k = \frac{1}{n} \sum_{j=1}^{n} d_{kj} u_\epsilon(x_k - x_j) \quad for \quad k = 1, ..., n \tag{2.11}$$

with

$$u_\epsilon(t) = \begin{cases} (t/\epsilon)(2 - |t|/\epsilon) & \text{if } |t| < \epsilon \\ sign(t) & \text{if } |t| \geq \epsilon \end{cases}$$

Finally, the $(r+1)^{th}$ iteration of the algorithm for minimizing $\sigma_\epsilon(X)$ is given by

$$x_k^{[r+1]} = \frac{1}{n} \sum_{j=1}^{n} d_{kj} u_\epsilon(x_k^{[r]} - x_j^{[r]}), \quad for \quad k = 1, ..., n \tag{2.12}$$

Pliner recommend a sequence of $\epsilon_i, i = 1, ..., N$ for minimizing $\sigma_{\epsilon_i}(X)$ over X such that $\epsilon_1 > \epsilon_2 > ... > \epsilon_N > 0$. The reason of choosing this sequence is that we can firstly indicate roughly the region that the global minimum belongs to and then find out the exact position of it by decreasing the value of ϵ. The starting value of ϵ should be chosen carefully because if ϵ_1 is too large, the range of the region will be too large while the smoothing effect will not be good enough if ϵ_1 is too small. Pliner suggest the starting value of ϵ_1 as $2max_{1 \leq i \leq n} n^{-1} \sum_{i=1}^{n} d_{ij}$ and $\epsilon_i = \epsilon_1(N - i + 1)/N$ for $i = 2, ..., N$. It is known that the larger N is, the smaller

value of $\sigma_\epsilon(X)$ will be found, but the computing time, as the same time, will be longer. Therefore, the choice of N is the trade-off of precision and computing time. The procedures for finding the coordinates of objects this time are:

1. Use $X^{[0]}$(can be random or not) as a starting configuration

2. Set i = 1

3. Set r = 1, Compute $\sigma(X^{[r-1]})$

4. Until $\sigma(X^{[r-1]}) - \sigma(X^{[r]}) < \epsilon$ or r = maximum number of iterations:

   Computer Pliner's smoothing algorithm$X^{[r+1]}$ by using ϵ_i,

   r = r + 1

5. $Z^{[i]} = X^{[r]}$ and $Z^{[i]}$ will be used as the starting value for the next minimization

6. $i = i + 1$, Goto 3 when $i \leq N$

7. The final solution, $Z = Z^{[N]}$

During the procedures, the value of ϵ_i becomes smaller after the updating which will make $\sigma_\epsilon(X)$ closer to $\sigma(X)$ and when $\epsilon \to 0$, $\sigma_\epsilon(X) \to \sigma(X)$ One of the advantages of using smoothing algorithm is that the final solution will not be affected when different starting configuration are used. That mean we can get the result by using a random $X^{[0]}$.

The computational experiments by Pliner suggested that there are more than 60% of the runs can achieve the global minima and sometimes 100% of term can achieve with suitable value of ϵ and N. Therefore, the smoothing approach is one of the important algorithm for finding global minima in unidimensional scaling problem. The result shown in Figure 1.5 is the unidimensional representation of the 7 kinds of crimes given in Table 1.3 by using Pliner's smoothing algorithm

2.3 Circular Unidimensional Scaling/Circumplex Model

The objective of circular unidimensional scaling(CDS) is to place the objects around a close continuum such that the minimum distance between the two objects, reflects originally proximity between them as well as possible, Hubert, Arabie and Meulman(1997) proposed a method of CDS to minimize the loss function

$$\sum_{i<j} (d_{ij} + c - min\{|x_j - x_i|, l - |x_j - x_i|\})^2 \tag{2.13}$$

where c is some constant to be estimated and l denotes the total length of the circle such that $min\{|x_j - x_i|, l - |x_j - x_i|\}$ gives the minimum length between object i and j. With a fixed permutation, $\varphi(.)$ such that $\varphi(i) = j$ if object j is placed at position i, the method is developed is developed with the following

setting:

Let

$$\Delta_{\varphi} \equiv \left\{ \{q_{ij} | q_{ij} = d_{\varphi(i)\varphi(j)} + c, \ for \ some \ constant \ c, i \neq j, q_{ii} = 0, 1 \leq i \leq n \right\}$$

and

$$\Xi \equiv \{ \{r_{ij} | r_{ij} = |x_j - x_i|, \ such \ that \ x_1 \leq x_2 \leq ... \leq x_n \}$$

Then, two arbitrary matrix with main diagonal of all zeros, $U = \{u_{ij}\}$ and $V = \{v_{ij}\}$ project onto Δ_{φ} and Ξ respectively. The detail of the projection process onto Δ_{φ} has been shown by Hubert, Arabie and Meulman (1997).

A summary measure of 'variance account for' was provided to measure the fitting of the model, where VAF is defined as follows

$$VAF = 1 - \left\{ \frac{\sum_{i<j} \left(d_{\varphi(i)\varphi(j)} + c - |x_j - x_i| \right)^2}{\sum_{i<j} \left(d_{ij} - \overline{d} \right)^2} \right\} \tag{2.14}$$

where $\overline{d}$ is the mean of the proximity values and the value of VAF is from zero to one.

Chapter 3

A new algorithm for CDS

In the last chapter, we have introduced the ideas and existing methods for dealing with MDS, UDS, CDS problems. In this chapter, we propose a better starting values for Guttman's updating algorithm and Pliner's smoothing algorithm for UDS. We also propose a new algorithm for CDS. In some cases, the CDS representation can provide a better solution(smaller objective function value) than UDS representation. Meanwhile, although the objects are represented in a circular way, it is still a unidimensional problem, which make the interpretations become easier. In particular, the proposed method may be suitable in many situations, especially for some psychological, social or marketing researches.

As mentioned earlier, the objective function may have many local minima and the problem will become more serious when the number of objects is large. Therefore, we will first suggest a method of choosing a good starting value for

both Guttman's updating method and Pliner's smoothing method. With this starting value, we may have a good chance to find a better solution or even the global minimum.

3.1 Method of choosing a good starting value in Guttman's updating algorithm and Pliner's smoothing algorithm

A random starting configuration is suggested by Guttman and Pliner for both updating algorithm and smoothing algorithm. However, the problem of being trapped in local minima for Guttman's updating method is very common if a random starting value is used and Pliner's method may also have this problem if n is very large. Therefore, it is a common practice to repeat the algorithm many times with randomly chosen starting value and pick the best among these solutions. Even when a global solution can be found, sometimes the number of iterations required is large. We now suggest a better method of choosing the starting value based on the observed distance matrix, d_{ij} of the objects.

We first find out the row in the distance matrix, $\mathbf{D} = (d_{ij})$ so that this row has the largest row sum among others(since the distance matrix is symmetric, it is equivalent to find out the column with maximum column sum)and then rank

each entry in that row in descending order. These ranks are then used as the starting value for the Guttman's updating or Pliner's smoothing algorithm.

Note that $\sum_{j=1}^{n} d_{ij}$ is the total distance of object i to other points, this value will be the largest if the true position of object i is at the first or at the end. Furthermore, once we fixed object i as the first object on the line, the distance of object j to i, d_{ij} reflect the closeness of object j to object i. For example, if row k has the largest row sum, it indicates that object k is located at the first position. The object with the shortest distance from object k is located at the second position. The positions of the other objects can then be found by using this method such that $d_{kO(1)} < d_{kO(2)} < ... < d_{kO(n)}$, where $O(i) = j$ if object j is at position i on the line. This starting value is not only appealing but also have the following advantage:

Theorem 3.1. *If the observed distance, d_{ij} is measured without error, i.e. $d_{ij} = |x_i - x_j|$, then the ture coordinates $x_1, ..., x_n$ can be obtained in one iteration by using Guttman's updating algorithm with the above proposed starting value.*

Proof

We divide the proof into two parts. We will first show that if object i is the first or the last object on a line, its total distance to other objects, $S_i = \sum_{j=1}^{n} d_{ij}$ is the largest. Then we show that the ture coordinates, $x_1, ..., x_n$, can be obtained in one iteration when the proposed starting value is used.

Suppose that there are n objects, $x_1, ..., x_n$ and $x_{(i)}$ is the i^{th} ordered object. Assume that there is an object, $x_{(k)}$, $k \neq 1, n$ such that $S_{(k)} > S_{(1)}$ and $S_{(k)} > S_{(n)}$. Consider

$$\begin{aligned}
& S_{(1)} + S_{(n)} - 2S_{(k)} \\
= & \left[\sum_{i=1}^{n} d_{(1)(i)} + \sum_{i=1}^{n} d_{(n)(i)}\right] - 2\sum_{i=1}^{n} d_{(k)(i)} \\
= & \; nd_{(1)(n)} - \left[d_{(1)(k)} + d_{(2)(k)} + ... + d_{(n)(k)} + d_{(k)(n)} + d_{(k)(n-1)} + ... + d_{(k)(1)}\right] \\
= & \; nd_{(1)(n)} - \left[d_{(1)(n)} + d_{(2)(n-1)} + ... + d_{(n-1)(2)} + d_{(n)(1)}\right] \\
> & \; 0, \qquad (3.1)
\end{aligned}$$

which is contradict to our assumption. Therefore, S_i is largest if the true position of object i is at the first or at the end. Once when we know the first or last object, we can obtain the rank of other objects by comparing their interpoint distance from it and use it as the starting value.

Now we show that the ture coordinates, $x_1, ..., x_n$ can be obtained in one iteration using Guttman's updating algorithm. Recall from (2.6), we can write that

$$x_{(i)}{}^{[2]} = \frac{1}{n}\sum_{k=1}^{n} d_{(i)k} sign(x_{(i)}{}^{[1]} - x_k{}^{[1]}). \qquad (3.2)$$

Suppose $x_{(j)} > x_{(i)}$, if we use the proposed starting value, then

$$x_{(i)}{}^{[2]} = \frac{1}{n}\left[\sum_{k=1}^{i-1} d_{(i)(k)} - \sum_{k=i+1}^{n} d_{(i)(k)}\right],$$

$$x_{(j)}{}^{[2]} = \frac{1}{n}\left[\sum_{k=1}^{j-1} d_{(j)(k)} - \sum_{k=j+1}^{n} d_{(j)(k)}\right].$$

$$\begin{aligned}
x_{(j)}{}^{[2]} - x_{(i)}{}^{[2]} &= \frac{1}{n}\left[\sum_{k=1}^{j-1} d_{(j)(k)} - \sum_{k=j+1}^{n} d_{(j)(k)} - \sum_{k=1}^{i-1} d_{(i)(k)} + \sum_{k=i+1}^{n} d_{(i)(k)}\right] \\
&= \frac{1}{n}\left[\sum_{k=1}^{i-1}\left(d_{(k)(j)} - d_{(k)(i)}\right) + \sum_{k=j+1}^{n}\left(d_{(i)(k)} - d_{(j)(k)}\right) + \right. \\
&\quad \left. \sum_{k=i+1}^{j-1}\left(d_{(i)(k)} + d_{(k)(j)}\right) + 2d_{(i)(j)}\right] \\
&= \frac{1}{n}\left[(i-1)d_{(i)(j)} + (n-j)d_{(i)(j)} + (j-i-1)d_{(i)(j)} + 2d_{(i)(j)}\right] \\
&= d_{(i)(j)}. \qquad (3.3)
\end{aligned}$$

Similarly we can prove for $x_{(j)} < x_{(i)}$, $x_j{}^{[2]} - x_j{}^{[2]} = -d_{ij}$. Therefore, $|x_j{}^{[2]} - x_j{}^{[2]}| = d_{ij}$ for all i, j and the proof is complete.

Since there are many solutions for a particular set of data can minimize the objective function(the coordinates may be shifted or reflected). We usually set a constraint to the solution such that they are centered(i.e. the sum of the coordinates is equal to zero, $\sum_{i=1}^{n} x_i = 0$). As shown in (2.8), the Guttman's updating algorithm as well as Pliner's smoothing algorithm has the self-centering property. Therefore, even though the starting value is not centered, the algorithm will make the solution satisfy the self-centering constraint.

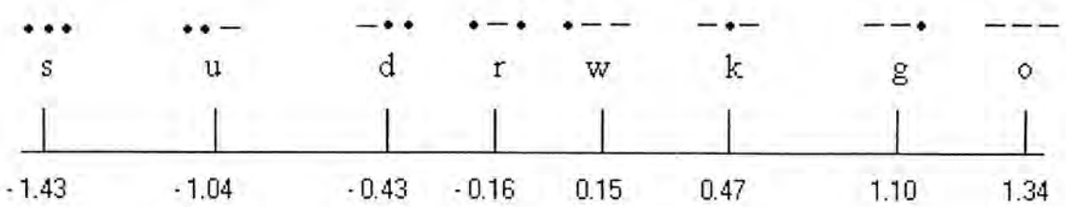
To illustrate our approach, we consider a simple 8×8 dissimilarity matrix for the Morse code symbols of eight alphabet, data from *Rothkof*(1957). Table 3.1 gives the distance matrix as well as the row total. The row corresponding to symbol 's' has the maximum row sum. The rank of that row, $z = (3, 7, 6, 8, 4, 1, 2, 5)$ is then used as the starting value.

Table 3.1: A proximity matrix for the eight Morse code symbols

Symbol	d	g	k	o	r	s	u	w	row total
d: - • •	0.00	1.22	0.46	1.85	1.32	1.41	1.51	1.51	9.28
g: - - •	1.22	0.00	1.42	0.42	1.53	1.90	1.74	1.43	9.66
k: - • -	0.46	1.42	0.00	1.43	1.57	1.85	1.33	1.38	9.44
o: - - -	1.85	0.42	1.43	0.00	1.78	1.91	1.83	1.49	10.71
r: • - •	1.32	1.53	1.57	1.78	0.00	1.66	1.37	0.77	10.00
s: • • •	1.41	1.90	1.85	1.91	1.66	0.00	0.93	1.76	11.42
u: • • -	1.51	1.74	1.33	1.83	1.37	0.93	0.00	1.47	10.18
w: • - -	1.51	1.43	1.38	1.49	0.77	1.76	1.47	0.00	9.81

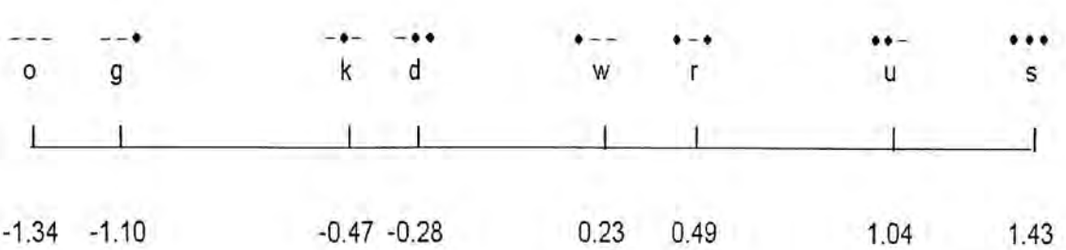
By using Guttman's updating algorithm with the starting configurations, the solution for linear unidimensional scaling was converged in one iteration and the corresponding order of the alphabet was obtained by sorting the co-ordinates of them in ascending order. The result are shown in Fig. 3.1

Figure 3.1: A unidimensional scaling representation



The value of the $\sigma(X)$ is 9.33 and the value of Stress-1, σ_1 is equal to 0.151. Since the observed distance, d_{ij} may contains errors or the true configuration may not be unidimensional, σ is greater than zero. To compare the Guttman's updating algorithm and Pliner's smoothing algorithm, we apply the Pliner's smoothing algorithm to the same data set and the result is shown in Figure 3.2.

Figure 3.2: A unidimensional scaling representation of Morse Code(by Pliner's smoothing algorithm)



The value of the $\sigma(X)$ is 8.24 and the value of Stress, σ_1 is equal to 0.133 when Pliner's smoothing algorithm is used. This indicates that Pliner's smoothing algorithm gives better solution (smaller $\sigma(x)$) than Guttman's updating algorithm.

It is because Guttman's updating algorithm is trapped in a local minimum. In some cases, as the above example shows, only a local minimum can be found by Guttman's updating algorithm in spite of using the proposed method as the starting value. Note also that in this solution, the order and coordinates of these alphabets reflect the similarity or dissimilarity of their morse codes.

To study the efficiency of the above proposed method as the starting value (we called this method as fixed starting value method hereafter), we generate n data, $(x_1, ..., x_n)$ from Uniform(0,1) and calculate the interpoint distances, $d_{ij} = |x_i - x_j|$ such that they have a perfect unidimensional structures(without any errors). Guttman's updating algorithm and Pliner's smoothing algorithm are applied to these data to compare their performance. The performance of using random and fixed starting value method will then be compared. We will also add $i.i.d.$ $Normal(0, \sigma_\tau^2)$ random error, τ_{ij} to the interpoint distance, d_{ij}. To prevent $d_{ij} = |x_i - x_j| + \tau_{ij} < 0$, we take $d_{ij} = |x_i - x_j| + |\tau_{ij}|$.

The convergence criterion set here was based on the absolute changes from one iteration to the next in calculating the Stress value being less than 10^{-5}; the same convergence criterion was used in both Guttman's updating algorithm and Pliner's smoothing algorithm. We restricted our study to $n = 100, 1000; \sigma_\tau = 0.5$. We first generate 5 data sets for each n and then generate 50 random starting values from Uniform(0,1). Finally, we compare the performance of Guttman's updating algorithm and Pliner's smoothing algorithm by using these random

starting values with the proposed fixed starting value. The maximum, minimum, the mean stress value and the average of the number of iterations required among these 50 solutions are reported. Results are shown in Table A.1 - Table A.5 of Appendix A.

From Table A.1, one can see that for both small and large n, if no error is added to the interpoint distance,d_{ij}, global minimum can be found when we apply the fixed starting value to Guttman's updating algorithm. However, even if we apply 50 random starting values for Guttman's updating algorithm on the same data set, not all of the trials can find the global minimum. When Pliner's smoothing algorithm is applied for the same data set, the global minimum can be found for both n = 100 and 1000 no matter random starting value or fixed starting value is used. This also demonstrates that Guttman's updating algorithm may be trapped in some local minima while Pliner's smoothing algorithm can escape from them.

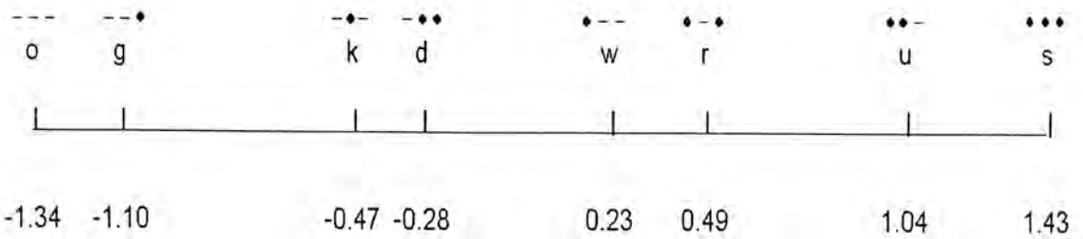
In Table A.2, when $n = 100$ and τ_{ij} is added to d_{ij}, using both fixed and random starting value cannot find the true coordinates of objects, but the stress value given by using fixed starting value is still similar to the best solution among those 50 random trials for each data set. We also notice that the solutions of Pliner's smoothing algorithm are better than that of Guttman's updating algorithm and they will not be affected when different starting values are used. The same conclusion can be drawn for $n = 1000$, as shown in Table A.3.

Table A.4-A.5 gives the average number of iterations required. The number of iterations required for Pliner's smoothing algorithm is more than that of Guttman's updating algorithm in all situations. Also note that when d_{ij} is measured without error, Guttman's updating algorithm gives the true solution in one iteration as stated and proved in Theorem 3.1.

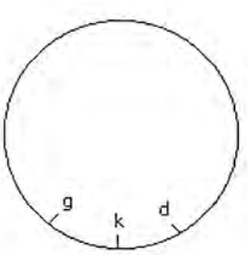
Based on the above simulation study, the fixed starting value method significantly improve the Guttman's updating algorithm. Furthermore, Pliner's smoothing algorithm gives better solution (smaller stress value) than Guttman's updating algorithm, but the number of iterations required in Pliner's smoothing algorithm is much bigger. Therefore we recommend Pliner's smoothing algorithm when n is small or when the computation time is inexpensive. Guttman's updating algorithm is preferred when n is large or when the computation time is expensive.

3.2 A new approach for circular unidimensional scaling

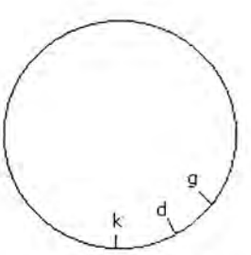
In this section, a new algorithm for CDS is proposed. We will illustrate this algorithm by the Morse code example in Table 3.1 again. First a UDS solution of these 8 objects is obtained as described in Section 3.1. The solution is given in Figure 3.2, we reproduce here for reference.



Then we need to construct a circular configuration for these 8 objects. Since $'k'$ and $'d'$ are the two closest objects ($d_{kd} = 0.19$) among others, we start with objects $'k'$ and $'d'$ as the left and right end of a circular configuration respectively.



When $\mathrm{d_{gk}} < \mathrm{d_{gd}}$



When $\mathrm{d_{gk}} > \mathrm{d_{gd}}$

Now we consider the problem of placing the next object, $'g'$ to the left side of $'k'$ or to the right side of $'d'$. Object $'g'$ should be placed on the left side of object $'k'$ when $d_{gk} < d_{gd}$(object $'g'$ is closer to $'k'$ than object $'d'$) and should be on the right side of $'d'$ if $d_{gk} > d_{gd}$ (object $'g'$ is closer to object $'d'$ than object $'k'$). By continuing this comparison, the order for all n objects, $O(.)$ can be obtained. Furthermore, the length of the circle, l can be estimated by $\widehat{l} = |y_{O(1)} - y_{O(n)}| + |y_{O(1)} - y_{O(2)}| + ... + |y_{O(n-1)} - y_{O(n)}|$ where y_i is the estimated

coordinates of object i obtained by Pliner's smoothing algorithm. The other way of estimating l is $\tilde{l} = d_{O(1)O(n)} + d_{O(1)O(2)} + ... + d_{O(n-1)O(n)}$. If the observed distance, d_{ij} contain no error and we can find out the true order of the objects, $\tilde{l}$ will be equal to l. However, if d_{ij} contain errors, the value of $\tilde{l}$ may be very different from the true length l even if the order is correct. We will study and compare these two estimation methods $\hat{l}$ and $\tilde{l}$ later in our simulation.

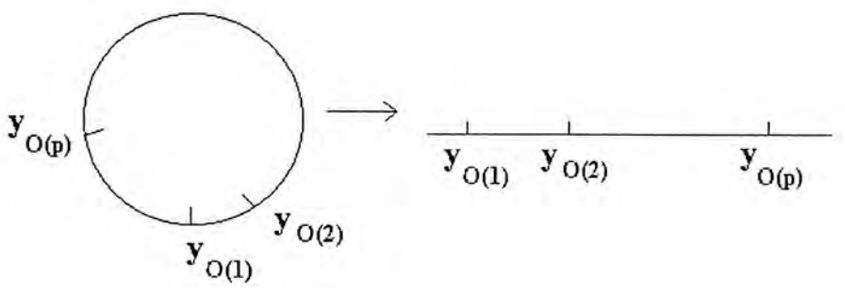
Once the circular order of these n objects and $\hat{l}$ are obtained, we can apply the Pliner's smoothing algorithm to improve the stress value. Since the interpoint distance defined in CDS is different from UDS, we should modify the original distance matrix first. Note that if $|y_i - y_j| > \hat{l}/2$, we have to change d_{ij} to $d'_{ij} = \hat{l} - d_{ij}$, i.e.,

$$d'_{ij} = \begin{cases} \hat{l} - d_{ij} & \text{if } |y_i - y_j| > \hat{l}/2 \\ d_{ij} & \text{otherwise} \end{cases} \tag{3.4}$$

This process allow us to cut the circle into a straight line and that $\mathbf{D}'$ is the distance matrix constructed from the linear configuration of the objects. For example, in Figure 3.3, we can see that $|y_{O(1)} - y_{O(2)}| < \hat{l}/2$, thus $d'_{O(1)O(2)} = d_{O(1)O(2)}$ and $|y_{O(1)} - y_{O(p)}| > \hat{l}/2$, so we take $d'_{O(1)O(p)} = \hat{l} - d_{O(1)O(p)}$.

After this modification, we apply Pliner's smoothing algorithm to $\mathbf{D}'$ to find the final solution. The formal description of the proposed method is summarized as follow:

Figure 3.3: Changing the configuration of the objects from circular to linear



1. Set z be the starting configuration, where z is the ranks of the elements in a row of $n \times n$ dissimilarity matrix, $\mathbf{D} = \{d_{ij}\}, i, j = 1, ..., n$ which has the maximum column sum.(i.e. use the fixed starting value method proposed in Section 3.1

2. Apply Pliner's smoothing algorithm and save the coordinates of the objects in y where $y = (y_1, ..., y_n)$.

3. Find out the two closest objects, i and j such that the value of $|y_i - y_j|$ is the smallest.

4. Calculate $|y_i - y_k|$ for all k and then sort and save the order of the objects to $p = (p_1, ..., p_n)$ according to these values where p is the vector used to save the linear order of the objects.

5. Set $w_i = y_{p_i}$ and $O_i = p_i$ for $i = 1, 2$ (where $w = (w_1, ..., w_n)$ and $O = (O_1, ..., O_n)$ are the vectors storing the circular coordinates and the orders of the objects respectively). Assign O_1 be the left point and O_2 be the right

point of the circle by setting $L = O_1$ and $R = O_2$.

6. Set iteration counter $k = 0$

7. Check from dissimilarity matrix, d. If $d_{Lp_{(k+3)}} \geq d_{Rp_{(k+3)}}$ (the distance of the new coming object from left, L is greater than that from right point, R), we update the next object, $p_{(k+3)}$ as the right point by setting

$$w_{(k+3)} = y_{(k+3)}, \; O_{(k+3)} = p_{(k+3)}, \text{ and } \; R = p_{(k+3)}.$$

Otherwise,we update $p_{(k+3)}$ as a left point by setting

$$w_{(n-k)} = y_{(k+3)}, \; O_{(n-k)} = p_{(k+3)}, \text{ and } \; L = p_{(k+3)}.$$

8. Increase iteration counter k by one. If $k \leq n - 3$, then go to 7.

9. Estimate the distance between the jth object and the $1st$ object for all j by computing

$$\hat{d}_{O_{(1)},O_{(j)}} = |w_{(1)} - w_{(j)}|, \qquad j = 2, ..., n$$

10. Estimate the total length of the circle, $\widehat{l}$ by

$$\widehat{l} = \widehat{d}_{O(1)L} + \widehat{d}_{O(1)R} + \widehat{d}_{LR}$$

11. Set $d'_{ij} = \widehat{l} - d_{ij}$ when $|y_i - y_j| > \widehat{l}/2$. Apply Pliner's smoothing algorithm again by using $O = (O_1, ..., O_n)$ as the starting value.

In this approach, Pliner's smoothing algorithm in the final step is needed since there is no guarantee that the solution is optimal in UDS is also optimal in CDS. This in fact can be shown by comparing the objective function of the two approaches(with and without final optimization) in the following example.

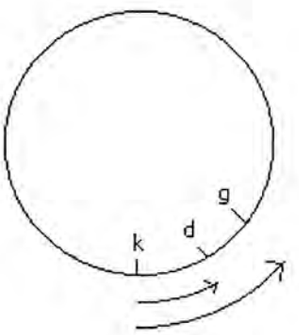
Example 3.1

Now, we illustrate our algorithm by considering the proximity matrix for eight Morse code in Table 3.1 again. As shown in Figure 3.2 before, a unidimensional scaling representation of the objects is obtained by Pliner's smoothing algorithm first. Since $'k'$ and $'d'$ are the two closest codes, we assign them on the first and second position and the linear order of the codes(start from $'k'$) is $(k\ d\ g\ w\ o\ r\ u\ s)$. Assuming that the circular order is anti-clockwise, we define $'k'$ as the left point and $'d'$ as the right point. The order of the third symbol $'g'$ is considered by comparing the interpoint distance between these three symbols, $'k'$,$'d'$ and $'g'$. From the dissimilarity matrix, $\mathbf{D}$, $d_{kg}(=\ 1.42) > d_{dg}(=\ 1.22)$, therefore $'g'$ is assigned to the right of $'d'$ and we update $'g'$ as the right point.

By comparing the interpoint distance as above, we obtained the circular order of all the objects, $O(.) = (k\ d\ g\ o\ s\ u\ r\ w)$. The value of $\widehat{l} = \widehat{d}_{ku} + \widehat{d}_{ks} + \widehat{d}_{su} = 3.80$. Based on the value of $\hat{d}_{ij}$ and $\hat{l}$, we construct the CDS representation of the Morse Code and the result is shown in Figure 3.5.

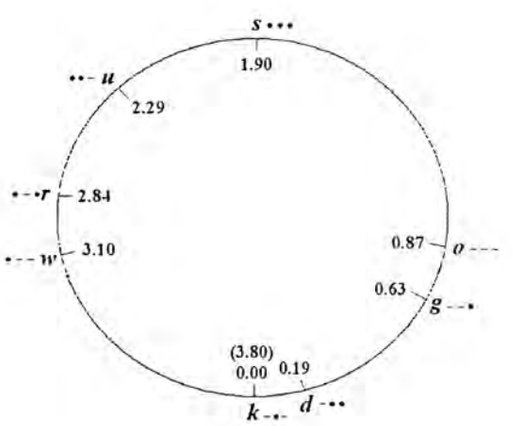
In this configuration, $\sigma_1 = 0.1338$. As mentioned before, the above solution is just optimal in linear case, but not necessary in circular case. Therefore, we

Figure 3.4: The choice of the position of 'g'



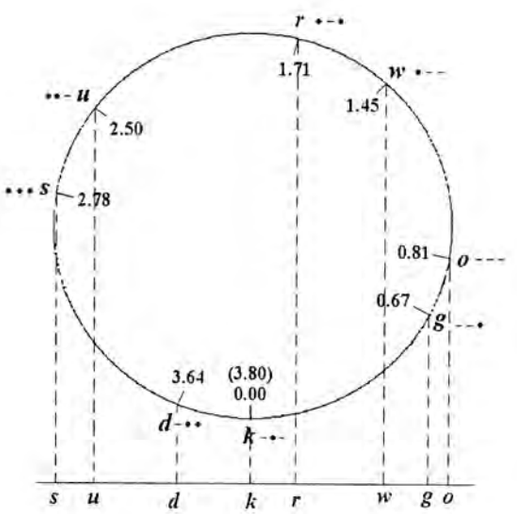
should perform a final optimization by applying Pliner's smoothing algorithm again.

Figure 3.5: The CDS of the Morse Code data from Table 3.1, $\sigma_1 = 0.1338$



After the Pliner's smoothing algorithm, the final solution is obtained and the stress value, σ_1 is 0.1209. The results shown in Figure 3.6. The coordinates of first position is set to 0.00 and the number in parenthesis, (3.8) is the total length of the circle. From Figure 3.6, we can see that the dashes are gently replaced by dots. After all of the dashes are replaced, there are a subsequence

Figure 3.6: The optimal representation of the Morse Code data from Table 3.1, $\sigma_1 = 0.1209$



of replacement of dots by dashes. It is also not surprise that the complement of three dashes(dots) are located at the opposite side, which indicates the great difference between them. Meanwhile, the projection of the objects on the line is different from the UDS configuration in Figure 3.2.

In order to study the performance of the proposed algorithm for CDS, we generate $(n+1)$ data, $(x_1, ..., x_{n+1})$ from Uniform(0,100). We use the first n^{th} data, $x_1, ...x_n$ as the coordinates and let x_{n+1} equal to the interpoint distance between $x_{O(1)}$ and $x_{O(n)}$, $d_{O(1)O(n)}$, so that $l = d_{O(1)O(2)} + d_{O(n-1)O(n)} + x_{n+1}$. We also add random error, $|\tau_{ij}|$ to d_{ij} where τ_{ij} are i.i.d. as $Normal(0, \sigma_\tau^2)$. We restricted our study to $n = 20$ and 50, $\sigma_\tau = 0.5$, 100 trial is made for each n. The results are shown in Table A.6-A.7 and Figure A.1-A.2 of Appendix A.

Table A.6 and A.7, we can see that when the distances, d_{ij} contains no error, the number of global minimum found by using $\tilde{l}$ is larger than that by using $\hat{l}$. When d_{ij} contains errors, we cannot obtain the global minimum and since the value $\tilde{l}$ is over-estimating from l and thus the solution is inferior to the solution obtained by using $\hat{l}$. The box plots in Figure A.1 - A.2 also show that the stress value by using $\tilde{l}$ is very sensitive to the random error. Therefore, it is better to use $\hat{l}$ instead of $\tilde{l}$ to estimate l for noisy data.

In our proposed algorithm, the object's order is a very important to the estimation of l as well as minimizing σ_1. For example, one can see from Table A.6 and A.7 that the value of σ_1 is close to zero if the order of objects is correct while σ_1 is close to 0.4 if the order is wrong.

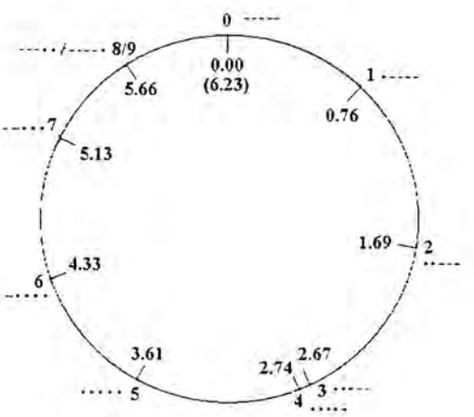
3.3 Examples

3.3.1 Comparison of the new approach to existing method

Recall the morse codes example from Section 1.3, we can see a circular shape from the configuration of the codes. Hubert, Arabie and Meulman(1997) used the proximity matrix in Table 1.6 to illustrate their method, the result is shown in Figures 3.7

We will apply our proposed method to the same data set and compare these two results. As an illustration of how this estimation proceeds for CDS, we first

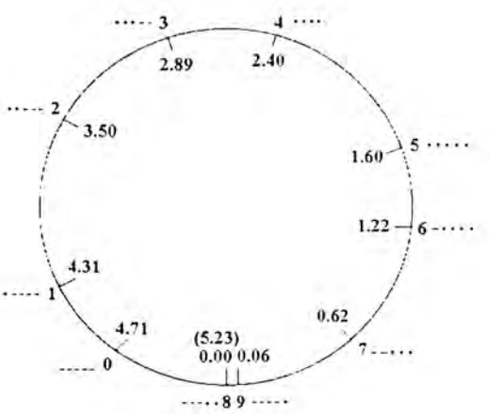
Figure 3.7: The CDS by Hubert, Arabie and Meulman's method for the Morse Code data from Table 1.6



find out the UDS solution of them and then decide the order of them by comparing d_{ij}. Finally, the total length of the circle is calculated and the optimization is achieved by applying Pliner's smoothing algorithm again. As is apparent in Figure 3.8, the order of these morse codes is almost the same as that in Figure 3.7 except the order of '8' and '9' are reversed. However, the difference is small because the coordinates of '8' and '9' in Figure 3.7 are the same while that in Figure 3.8 are very similar.

To compare the fitting of the two methods, we calculate the Stress value for both solutions. The value of σ_1 obtained by Hubert, Arabie and Meulman is 0.166 while our new method gives $\sigma_1 = 0.080$. In both figures, the permutation provides our interpretable ordering of the Morse Code symbols involving a regular replacement of dashes by dots moving clockwisely(Figure 3.7)/anti-

Figure 3.8: The CDS by proposed method for the Morse Code data from Table 1.6



clockwisely(Figure 3.8) until the symbol containing all dots is reached and then a subsequence replacement of the dots by dashes until the initial symbol containing all dashes is reached.

3.3.2 Illustrations of application to political data

A number of World leaders and politicians during Second World War are rated on a nine-point scaling from 1(very similar) to 9(very dissimilar)in order to judge the dissimilarities of them(Data from Everitt and Rabe-Hesketh (1997)). Table 3.2 shows the rating of them. We apply both the classical scaling and the new method to the data set. The solution of CDS by two methods are given in Figure 3.9 and Figure 3.10.

The stress value of the classical scaling and the proposed method are 0.066

Table 3.2: A proximity matrix for twelve Second World War politicians

Politician	1	2	3	4	5	6	7	8	9	10	11	12
1 Hitler(HI)	0											
2 Mussolini(MU)	3	0										
3 Churchill(CH)	4	6	0									
4 Eisenhower(EI)	7	8	4	0								
5 Stalin(ST)	3	5	6	8	0							
6 Attlee(AT)	8	9	3	8	9	0						
7 Franco(FR)	3	2	5	7	6	7	0					
8 De Gaulle(DE)	4	4	3	5	6	5	4	0				
9 Mao Tse-tung(MA)	8	9	8	9	6	9	8	7	0			
10 Truman(TR)	9	9	5	4	7	8	8	4	4	0		
11 Chamberlain(CM)	4	5	5	4	7	2	2	5	9	5	0	
12 Tito(TI)	7	8	2	4	7	8	3	2	4	5	7	0

and 0.114 respectively. It is not difficult to see that the stress value of classical scaling is smaller than our method as all of the coordinates are not restricted on the circular continuum. A possible interpretation for the vertical dimension in Figure 3.9 is that it represents the politicians' democratic credentials, where the positions of the moderates such as Attlee and Churchill are below the x-axis while that of the dictators, such as Hitler and Mussolini, are above the x-axis. However, it is not an easy task to interpret the findings in the horizontal dimension. A similar interpretation can also be made by the CDS representation in Figure 3.10 where we can see that the locations of the dictators are near to each others and it is also true for the moderates.

Figure 3.9: The MDS representation by classical scaling method for the similarity of the politicians from Table 3.2

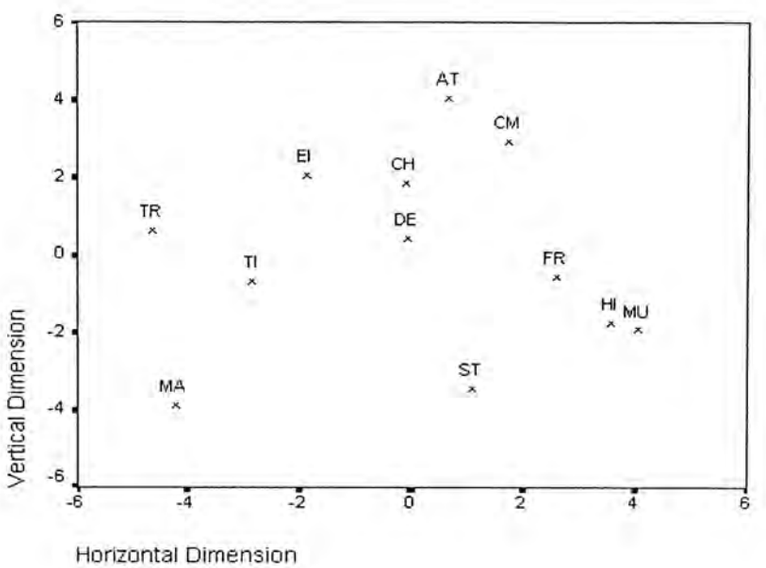
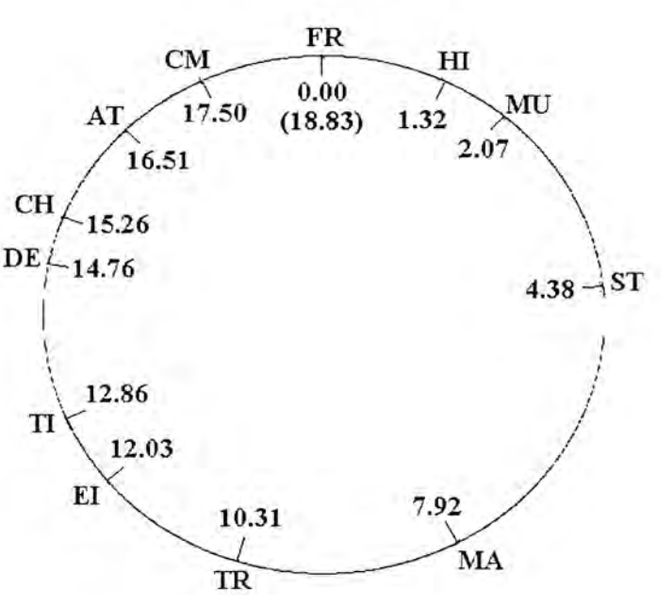


Figure 3.10: The CDS representation by proposed method for the similarity of the politicians from Table 3.2



Chapter 4

Conclusion and Extensions

In this thesis, we first introduced the UDS, MDS and CDS problems and discussed some existing approaches to such problems. Then we pointed out the local minimum problem that we may face when we apply the Guttman's updating algorithm or Pliner's smoothing algorithm to UDS problems. We proposed a modification of choosing a starting configuration for such techniques. Through a simulation study, we showed that this modification can produce a better solution(smaller stress value) and requires less iteration to converge. Furthermore, we proposed a new approach based on the Pliner's smoothing algorithm for the circular unidimensional problem. Simulation study is carried out to study the performance of this new method and indicates that the true coordinates can be found by the proposed method if the interpoint distances are measured without error.

In this chapter, we would like to discuss some extensions of UDS to Data Mining. Roughly speaking, data mining refers to the automatic extraction of knowledge, patterns and relationships from huge databases. With the growth of technology in hardware, large amount of data can be accumulated, typical examples are bank customer's database, supermarket's scanner database, credit cards transaction records...etc. These huge databases are usually measured in terabyte and is extremely difficult to analyze.

One natural question in data mining is to find groups in the data. For example, we want to group customer with similar background into clusters or we may want to group the credit card holders with similar spending patterns...etc. Usually cluster analysis, especially K-mean clustering method is used to find these groups. However, there are some limitations of K-mean clustering method. For example, the number of cluster has to be specified in advance, the grouping is sensitive to the initial seed...etc. The UDS algorithm provides an alternative method for finding similar groups in the data. We briefly describe two possible extensions.

1. **Clustering the objects (observations)**

   First we defined a distance metric (e.g. Euclidean distance) between two objects. Then the Guttman's updating algorithm or the Pliner's smoothing algorithm with fixed starting value can be used to arrange these objects in a linear order with coordinates assigned to each objects. Similar objects should have similar coordinates. A large jump or gap in the coordinate may

indicate a possible boundary of two different clusters.

2. **Clustering the variables (features)**

   In data mining, it is not uncommon to have hundreds of variables in the database. We may want to group similar variables into clusters to reduce the dimension as in principle component analysis or factor analysis. A possible way is to apply UDS on the variables. First we defined a dissimilarity (distance) between variables. For example, the dissimilarity (distance) between variables i, j can be defined as $d_{ij} = 1 - |r_{ij}|$, where r_{ij} is the correlation coefficient of variables i and j. Then the UDS method will arrange the variables in a linear order with similar variables having similar coordinates. Again a large jump or gap in the coordinate may indicate a possible boundary of two clusters.

Once the objects or the variables are arranged in a linear order, an automatic procedure for finding gaps is needed. This problem has been well studied and called the 'change point problem' in the statistical literature. Since the objects or variables has been arranged in a linear order, formal hypothesis testing procedure can be derived to detect possible gaps in the objects or variables. These possible extensions are not been studied in detail in this thesis, we leave these issues for future research.

Appendix A

Figures and Tables

Figure A.1: The box plots of the stress value given by the proposed method,(n=20, 100 trials, $\sigma_\tau = 0.5$)

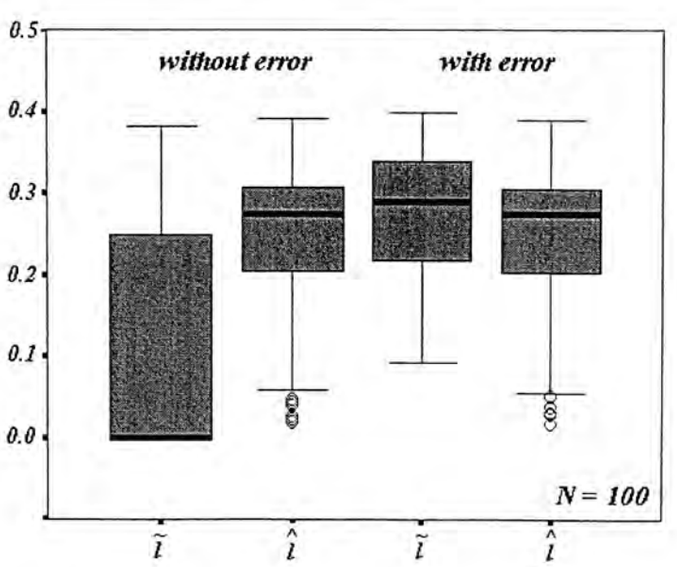


Figure A.2: The box plots of the stress value given by the proposed method,(n=50, 100 trials, $\sigma_\tau = 0.5$)

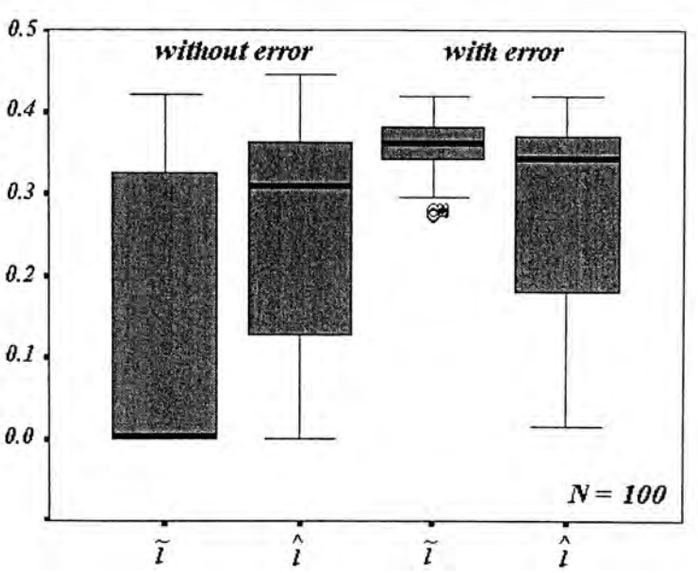


Table A.1: The comparison of the stress value of Guttman's updating algorithm on data without noise (50 random starting values)

	n = 100			n = 1000	
Simulation		Random	Fixed	Random	Fixed
1	Stress Value	min = 0.0000 max = 0.4305 mean = 0.1948 S.D. = 0.1638	0.0000	min = 0.0000 max = 0.4001 mean = 0.1575 S.D. = 0.1600	0.0000
2		min = 0.0000 max = 0.4466 mean = 0.1586 S.D. = 0.1594	0.0000	min = 0.0040 max = 0.3860 mean = 0.1169 S.D.= 0.1114	0.0000
3		min = 0.0000 max = 0.4196 mean = 0.2395 S.D. = 0.1406	0.0000	min = 0.0000 max = 0.3978 mean = 0.1182 S.D. = 0.1407	0.0000
4		min = 0.00000 max = 0.4077 mean = 0.1769 S.D. = 0.1228	0.0000	min = 0.0118 max = 0.3945 mean = 0.1339 S.D. = 0.1089	0.0000
5		min = 0.0000 max = 0.3807 mean = 0.2132 S.D. = 0.1290	0.0000	min = 0.0000 max = 0.3971 mean = 0.1752 S.D. = 0.1471	0.0000

Table A.2: The comparison of the stress value of Guttman's updating algorithm and Pliner's smoothing algorithm for n=100 and $\sigma_\tau = 0.5$ (50 random starting values)

	Guttman			Pliner	
Simulation	Random		Fixed	Random	Fixed
1	Stress Value	min = 0.1660	0.1654	min = 0.0913	0.0913
		max = 0.3335		max = 0.0913	
		mean = 0.3025		mean = 0.0913	
		S.D. = 0.0430		S.D. = 0.0000	
2		min = 0.1663	0.1663	min = 0.0938	0.0938
		max = 0.3366		max = 0.0938	
		mean = 0.2829		mean = 0.0938	
		S.D. = 0.0611		S.D. = 0.0000	
3		min = 0.1619	0.1620	min = 0.0883	0.0883
		max = 0.3405		max = 0.0883	
		mean = 0.3043		mean = 0.0883	
		S.D. = 0.0427		S.D. = 0.0000	
4		min = 0.1887	0.1798	min = 0.1111	0.1111
		max = 0.3398		max = 0.1111	
		mean = 0.2977		mean = 0.1111	
		S.D. = 0.0379		S.D. = 0.0000	
5		min = 0.1734	0.1735	min = 0.1019	0.1019
		max = 0.3281		max = 0.1019	
		mean = 0.2904		mean = 0.1019	
		S.D. = 0.0451		S.D. = 0.0000	

Table A.3: The comparison of the stress value of Guttman's updating algorithm and Pliner's smoothing algorithm for n=1000 and $\sigma_\tau = 0.5$ (50 random starting values)

	Guttman			Pliner	
Simulation	Random		Fixed	Random	Fixed
1	Stress Value	min = 0.1660 max = 0.3335 mean = 0.3025 S.D. = 0.0430	0.1654	min = 0.0981 max = 0.0981 mean = 0.0981 S.D. = 0.0000	0.0981
2		min = 0.1663 max = 0.3366 mean = 0.2829 S.D. = 0.0611	0.1663	min = 0.1003 max = 0.1003 mean = 0.1003 S.D. = 0.0000	0.1003
3		min = 0.1619 max = 0.3405 mean = 0.3043 S.D. = 0.0427	0.1620	min = 0.0985 max = 0.0985 mean = 0.0985 S.D. = 0.0000	0.0985
4		min = 0.1887 max = 0.3398 mean = 0.2977 S.D. = 0.0379	0.1798	min = 0.1023 max = 0.1023 mean = 0.1023 S.D. = 0.0000	0.1023
5		min = 0.1734 max = 0.3281 mean = 0.2904 S.D. = 0.0451	0.1735	min = 0.0976 max = 0.0976 mean = 0.0976 S.D. = 0.0000	0.0976

Table A.4: The comparison of the mean number of iteration required by Guttman's updating algorithm and Pliner's smoothing algorithm for n = 100 (50 random starting values)

	Without error				With error			
	Guttman		Pliner		Guttman		Pliner	
Simulation	Random	Fixed	Random	Fixed	Random	Fixed	Random	Fixed
1	19.92	1.00	113.40	95.00	25.32	16.00	193.20	165.00
2	17.52	1.00	147.40	126.00	27.30	14.00	268.50	236.00
3	16.68	1.00	132.54	114.00	22.74	9.00	265.00	237.00
4	20.16	1.00	221.64	205.00	24.64	12.00	291.84	266.00
5	18.88	1.00	164.80	143.00	27.14	12.00	263.80	232.00

Table A.5: The comparison of the mean number of iteration required by Guttman's updating algorithm and Pliner's smoothing algorithm for n = 1000 (50 random starting values)

	Without error				With error			
	Guttman		Pliner		Guttman		Pliner	
Simulation	Random	Fixed	Random	Fixed	Random	Fixed	Random	Fixed
1	35.00	1.00	172.14	148.00	102.60	19.00	290.80	255.00
2	33.32	1.00	208.14	178.00	101.34	20.00	341.20	295.00
3	33.66	1.00	174.70	150.00	99.92	19.00	300.04	263.00
4	34.66	1.00	202.30	174.00	105.48	18.00	323.50	281.00
5	46.58	1.00	172.10	145.00	96.78	22.00	281.20	241.00

Table A.6: The stress value of the proposed algorithm for the data(in 100 trials) without and with error,$\sigma_\tau = 0.5$, n=20

		Without error		With error	
		$\tilde{l}$	$\hat{l}$	$\tilde{l}$	$\hat{l}$
Stress value	min	0.0000	0.0179	0.0926	0.0202
	1^{st} quartile	0.0005	0.2030	0.2164	0.2017
	median	0.0007	0.2739	0.2894	0.2741
	3^{rd} quartile	0.2477	0.3074	0.3362	0.3030
	max	0.3734	0.3901	0.3808	0.3913
	mean	0.1113	0.2453	0.2741	0.2500
	S.D.	0.1417	0.0969	0.0792	0.0943

Table A.7: The stress value of the proposed algorithm for the data (in 100 trials) without and with error,$\sigma_\tau = 0.5$, n=50

		Without error		With error	
		$\widetilde{l}$	$\hat{l}$	$\widetilde{l}$	$\hat{l}$
Stress value	min	0.0000	0.0006	0.2738	0.0159
	1^{st} quartile	0.0003	0.1180	0.3417	0.1703
	median	0.0005	0.3085	0.3619	0.3428
	3^{rd} quartile	0.3212	0.3613	0.3819	0.3686
	max	0.4213	0.4451	0.4200	0.4200
	mean	0.1443	0.2527	0.3599	0.2842
	S.D.	0.1619	0.1286	0.0319	0.1225

Appendix B

References

1. Borg, I. & Groenen, P. (1997), Modern multidimensional scaling: theory and application, Springer-Verlag, New York, Inc

2. Everitt, B. S. & Rabe-Hesketh, S. (1997), The analysis of proximity data, London: Arnold

3. de Leeuw, J. & Heiser, W, J. (1980), Multidimensional scaling with restrictions on the configuration, in Multivariate Analysis, Ed, P.R. Krishnaiah, Amsterdam: North- Holland, Vol. V, 501-522

4. Gower, J. C. (1966), Some distance properties of latent root and vector methods used in multivariate analysis, Biometrika, 53, 325-338

5. Green, P. E. & Rao, V. R. (1972), Applied multidimensional scaling: A comparison of approaches and algorithms, New York: Holt, Rinehart &

Winston

6. Green, P. E., Carmone, F. J. & Smith, S. M. (1989), Multidimensional scaling: concepts & applications, London: Allyn & Bacon

7. Guttman, L. (1968), A general nonmetric technique for finding the smallest coordinate space for a configuration of points, Psychometrika, 33, 469-506

8. Hubert, L. J., Arabie, P. & Meulman, J. (1997), Linear and circular unidimensional scaling for symmetric proximity matrices, British Journal of Mathematical and Statistical Psychology, 50, 253-284

9. Kendall, D.G. (1970), A mathematical approach to seriation, Philosophical Transcations of the Royal Society of London, 269, 125-35

10. Kruskal, J. B. (1964a) Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis, Psychometrika, 29, 1-27

11. Kruskal, J. B. & Wish, M. (1978), Multidimensional scaling, Newbury, CA: Sage

12. Pliner, V. (1984), A class of metric scaling models, Automation and Remote Control, 45, 783-788

13. Pliner, V. (1996), Metric unidimensional scaling & global optimization, Journal of Classification, 13, 3-18

14. Plutchik, R. & Conte, H. R. (Eds) (1997), Circumplex models of personality and emotions, Washington, D.C.: American Psychological Association

15. Restle, V. (1959), A metric and an ordering on sets, Psychometrika, 24, 207-220

16. Rothkopf, E. Z. (1957), A measure of stimulus similarity and errors in some paired- associate learning tasks, Journal of Experimental Psychology, 53, 94-101

17. Schiffman, S. S., Reynolds, M. L. & Young, F. W. (1981), Introduction to multidimensional scaling: theory, methods and applications, New York: Academic Press

18. Wilkinson, L. (1990), SYSTAT: The system for statistics, Evanston, IL: SYSTAT Inc

19. Yudin, D. B. (1965), Quantitative analysis of complex system I, Engineering Cybernetics, No. 1, 1-9